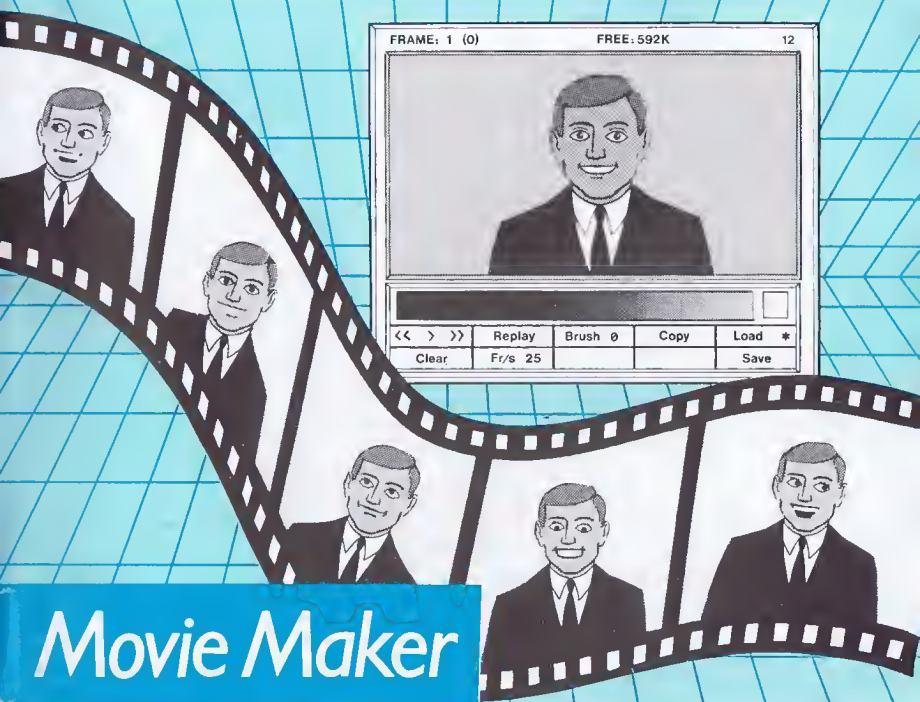


Volume 2
Issue 2

December
1988

Price £1.20

RISC USER



Movie Maker

THE MAGAZINE AND SUPPORT GROUP
EXCLUSIVELY FOR USERS OF THE ARCHIMEDES

CONTENTS

FEATURES

News	4
Getting to know Arthur	7
Public Domain Software for the Arc	29
Archimedes Visuals	38
Introducing ARM Assembler (Part 8)	42
Postbag	51
Hints & Tips	53

UTILITIES AND APPLICATIONS

Mouse Menu	11
Movie Maker	21
RISC User Toolbox (Part 6)	33

REVIEWS

ProArtisan Overview	5
The ABC Basic Compiler	17
Pipedream Spellcheck	30
Studio 24 Plus	47

ADVERTISERS

Computer Concepts 5, Healthdata 10, Micro Studio 10, Quercus Computer Systems 10, Ace Computing 15, Dabs Press 16, Beard Technology 20, Texcellence 20, Smtron 25, Clares Micro Supplies 26, Colton Software 32, Intelligent Interfaces 37, Norwich Computer 40, Gem Electronics 41, Computer Assisted Learning 46, Cambridge Microsystems 46, David Pilling 46, Silicon Vision 50, Archeffect 52, Data Store 52, P.A.J. Taylor 54, BEEBUG 54, 55

RISC User is published by BEEBUG Ltd.

Co-Editors: Mike Williams, Dr Lee Calcraft

Assistant Editor: Kristina Lucas

Technical Editor: David Spencer

Advertising: Sarah Shrive

Subscriptions: Mandy Milleham

BEEBUG, Dolphin Place, Holywell Hill, St.Albans,
Herts AL1 1EX. Tel. St.Albans (0727) 40303

All rights reserved. No part of this publication may be reproduced without prior written permission of the Publisher. The Publisher cannot accept any responsibility whatsoever for errors in articles, programs, or advertisements published. The opinions expressed on the pages of this journal are those of the authors and do not necessarily represent those of the Publisher, BEEBUG Limited.

BEEBUG Ltd (c) 1988



The Archimedes Magazine and Support Group.

EDITORIAL

As we report in our news page, the Micro User show held at Westminster last month was very well attended, and the Archimedes is beginning to make its mark, with new releases of software and hardware regularly announced. Those whose purchase of an Archimedes was an act of faith are at last being rewarded not only by the machine's native capabilities, but by some of the excellent software becoming available for it.

Perhaps best of all, the imminent arrival of the RISC OS operating system will place great power in the hands of all Arc owners. We understand that for a limited period the price of the upgrade will be just £29. For this price you will not only get a very powerful operating system with many advanced features, including multi-tasking through the WIMP, but you will also receive bundled applications on two discs. These will include a text processor, and a high quality object-based drawing package. If you are still in doubt, then perhaps the many other improvements on Arthur 1.2, such as the vastly increased screen loading and saving speed, may well sway you.

At such a reasonable upgrade price, we would urge all Arc users to take advantage of the offer at the earliest opportunity - though as you will probably know, the lead time for putting RISC OS into ROM means that it will not be available until April. Of course, all new machines will be shipped with the new operating system as soon as it becomes available.

Another offer which is worth considering carefully is the Volume One Special. The offer price of £4.95 runs out at the end of December. On the 1st of January the price will be £12.50 - but even at this price, we believe that the disc represents extremely good value.

The next issue of RISC User will be a joint issue covering the months of January and February - and with that may we wish a happy Christmas to one and all.

*This month's telesoftware password is **courtyard**.
(see BEEBUG pages on Micronet)*

SHOW SUCCESS

After the relatively poor attendance at last May's Electron and Micro User Show, many exhibitors must have been a bit apprehensive about November's show. However, these fears were ungrounded, and the attendance was very good. There are a number of possible reasons for this. Firstly, Christmas is fast approaching and an event like a show is the perfect opportunity to buy a few gifts. Secondly, the lack of an Acorn User show this year has meant that users have had to wait six months for this show, and finally, it may well be evidence of the growing interest in the Archimedes that prompted people to attend. Whatever the reasons, such a good attendance is very welcome, and hopefully it will continue in the future.

COLOUR IMAGES

Lingenuity has just released a podule that allows the Watford Electronics video digitiser podule to handle colour images. The new device, called Colour Converter accepts the PAL output available from most colour video cameras, and produces an output which can then be processed by the Watford Digitiser. The system works by taking the PAL colour signal and splitting it into its red, green and blue constituent images. Each of these can then be switched under software control into the Watford Digitiser. The mouse driven software supplied with the Colour Converter allows pictures to be grabbed, rescaled and saved. The Colour Converter costs £195.44 inc. VAT, and is available from Lingenuity, Lindis International Ltd., Wood Farm, Linstead Magna, Halesworth, Suffolk IP19 0DU, tel. (0986) 85476.

CONTROLLABLE CONFIGURES

Also new from Lingenuity (see above) is a WIMP-based configuration program, called Control Panel, which allows the configuration of your Archimedes to be changed by making selections from a series of icons. It is also possible to save and load sets of configuration data. Control Panel costs £17.19 inc. VAT from Lingenuity at the address given above.

CARING FOR YOUR ARC

BEEBUG has produced three products to make your Arc's life easier. The first is a two-piece dust cover which will keep the computer, keyboard and standard monitor

clean. This costs £8.55. The other two items are a one metre coiled keyboard extension cable at £7.95, and a keystrip holder for £2.95. All prices include VAT and apply to RISC User and BEEBUG members only. More details can be found in the Christmas retail catalogue accompanying this issue.

MORE MODE 15 ART

Pax from Z&Z Software is the latest 256 colour art package to be released for the Archimedes. Pax offers all the normal features, such as variable brush size, an air brush, and the ability to draw various shapes. Sprites can be pasted into the picture, and this allows the inclusion of images snatched from other programs etc.. Pictures created in Pax can include text in either the standard font or one of Acorn's 'fancy fonts'. Pax costs £15.95 inc. VAT, and can be obtained from Z&Z Software, Brecklands, Broad Oak, Shrewsbury, Shropshire SY4 3AH.

DISCS FROM COMPUTER CONCEPTS

For those users who do not have a ROM podule to run Computer Concepts' software, the company has released the Inter series and Spell-Master on disc for the Archimedes. Inter-Word and Inter-Sheet cost £33.35 each, Inter-Chart costs £21.85, and Spell-Master is £44.85. All prices include VAT. The disc software still has to be run under the 6502 emulator, and the ROM-LINK system for transferring data between packages isn't implemented. More details can be obtained from Computer Concepts, Gaddesden Place, Hemel Hempstead, Herts HP2 6EX, tel. (0442) 63933.

UTILITIES FOR PROGRAMMERS

Acorn has released its *Software Developers' Toolbox* for the Archimedes. This package, which requires at least 1Mbyte of memory, is really a pot-pouri of utilities. The main one being a symbolic debugger which allows the execution of programs written in languages such as ANSI C and Fortran 77 to be monitored and controlled. The other utilities include file compares and a memory tester, together with a number of commands to make the linking of compiled languages and machine code routines easier. The *Software Developers' Toolbox*, which is supplied on two discs together with two manuals, costs £228.85 inc. VAT, or £217.41 inc. VAT to RISC User and BEEBUG members.

RU



ProArtisan

An overview by Mike Williams

Those of you who were able to attend the PC Show at Earls Court in September will no doubt have been fascinated by the exquisite results obtained from Clares' professional art package, ProArtisan. We anticipate that the complete package will be available by the time you receive this issue of RISC User, and we expect to feature a full and comprehensive review in our next issue.

Initially, ProArtisan will be supplied in a polished wood artist's palette box as an added attraction, but it is the contents which will come under scrutiny as we endeavour to establish if ProArtisan is worth its not inconsiderable price.

In the meantime, we have been able to obtain a pre-release copy of ProArtisan and we reproduce here some of the images which are likely to become quite well known in the ensuing months.



ProArtisan is clearly an evolutionary development of Clares' original Artisan package, which will continue to be sold. ProArtisan uses the 256 colour modes (principally mode 15), and the default palette. You can choose any of the 256 colours from cleverly arranged 'paint charts' in which shades of similar colours are arranged together, rather like the paint charts you can consult when decorating your house.

Many of the features of Artisan have been retained, and anyone who has used the original



package will find much that is familiar. However, the use of 256 colours has allowed even Clares' carefully designed icons and screen displays to be enhanced even further.

Some of the features of ProArtisan

Colour Sets - a range of 16, 12 or 8 shades of a colour which may be used for graduated fills.

Colour Wash - an application of anti-aliasing which gives an effect like that resulting from drops of water falling on a painting.

Bezier Curves - producing smooth controllable free-style curves, which can also be used for tracing around irregularly shaped sprites.

Global Toner - reduces a picture to shades of one colour including old style sepia effects.

Distort - allows images to be distorted in ways that adds a 3 dimensional aspect to objects.

Printer Dumps - to cope with the 256 colour modes for output to a variety of printers including the latest 24 pin models and various colour printers.

*ProArtisan (£169.95 inc VAT)
is supplied by Clares Micro Supplies,
98 Middlewich Road, Rudheath, Northwich,
Cheshire CW9 7DA. Tel. (0606) 48511.*

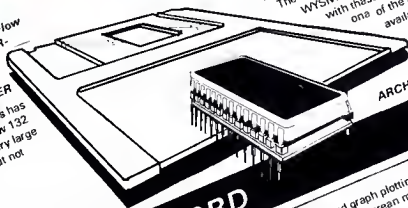
RU

Software for the Archimedes now available on disc...

Computer Concepts is now releasing new low cost DISC versions of its popular INTER-WORD, SHEET, CHART and SPELL-MASTER products for the Archimedes.

A simple to use but powerful spreadsheet. This has also been altered to take advantage of the new 132 column x 64 line screen modes allowing very large sheets to be viewed on screen. Easy to use but not lacking in facilities.

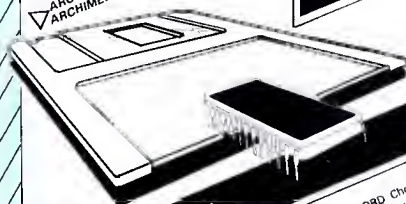
▽ ARCHIMEDES DISC £29.00 + VAT
ARCHIMEDES RDM £39.00 + VAT



INTER-WORD

A chart and graph plotting program, now supporting the 256 colour screen modes which can produce impressive colour charts and graphs. An ideal complement for INTER-SHEET, although it can plot graphs from practically any data.

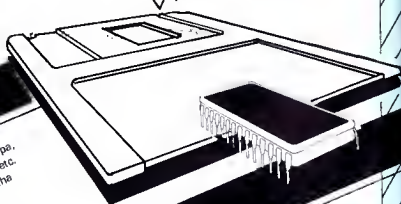
▽ ARCHIMEDES DISC £19.00 + VAT
ARCHIMEDES RDM £25.00 + VAT



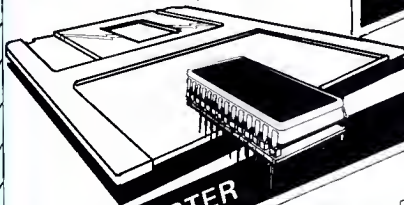
INTER-SHEET

An ideal partner for INTER-WORD. Check as you type, new intelligent guess routines, crossword solving etc. This is a native ARM product, and does not need the 6502 emulator.

▽ ARCHIMEDES DISC £39.00 + VAT
ARCHIMEDES RDM £51.30 + VAT



INTER-CHART



SPELL-MASTER

The INTER products run under the 6502 emulator but with some enhancements and other changes to suit the Archimedes. They are all compatible with the BBC versions and can load BBC files with no changes. No RDM-LINK facilities are available.



Computer Concepts Ltd

Gaddesden Place,
Hemel Hempstead, Herts. HP2 6EX
Telephone: 0442 63933 Fax: 0442 231632

Access/Bercleycard welcome.

Getting to Know Arthur (Part 1)

Whether you have used a BBC micro before or not, it is very easy to overlook many of the good features of Arthur, the Archimedes' operating system.

Mike Williams explains.

Arthur, as most Archimedes owners will know, is the name by which the Archimedes' operating system is known. And Arthur has many useful functions, if you know how to use and access them. If you have used a BBC micro previously, you will already be familiar with star commands (so-called because they are all preceded by a *star* or *asterisk* when entered from Basic). While most of these are still recognised by Arthur, it is easy to ignore the fact that there are many new commands as well.

The purpose of this short series is to look at some of the new functions provided by Arthur, both for those who have upgraded from BBC micros and for those for whom the Arc is their first Acorn micro.

It is perhaps tempting to think of all star commands as being operating system commands but this is not entirely correct. The name Arthur more precisely refers to the core operating system of the Archimedes which is then extended by a series of modules. These modules cover such features as the Window Manager, the Sound System, Sprites and the Filing System. All of this is resident in ROM, which means that all these functions are available on power-up. In addition, relocatable modules may be loaded from disc to extend the operating system even further, and add additional star commands to the system. Any modules, whether resident or relocatable, may be disabled if required.

In this article we will be looking predominantly at Arthur itself; later articles may deal with the star commands recognised by other resident modules. In addition, there is one star command (FX) which gives access to many different subroutines within the operating system, although these are often better accessed from machine code programs (via corresponding codes called SWI calls), or from Basic (using the keyword SYS to perform the same function). And if that were not enough, the VDU call used in Basic, and indeed many other Basic keywords, also access operating system routines.

We shall confine ourselves to those operating system functions accessed via star commands, and this will include a look at FX calls. So that you know what we are talking about, the basic operating system commands we shall cover are listed in Table 1.

*CONFIGURE	*KEY
*ECHO	*SET
*ERROR	*SETEVAL
*EVAL	*SETMACRO
*FX	*SHADOW
*GO	*SHOW
*GOS	*STATUS
*HELP	*TIME
*IF	*TV
*IGNORE	*UNSET

Table 1. Operating system commands

CONFIGURING THE SYSTEM

One chore which all Archimedes owners encounter is that of reconfiguring the system. This arises from the need to allocate different amounts of RAM for different circumstances, but also enables a wide variety of characteristics to be customised for when you switch the machine on, or press Ctrl-Break. You must also remember to press Ctrl-Break after reconfiguring the system in order to activate the change of setting (and thus the command cannot normally be used from within a program).

For example, sprites, screen displays, and fonts all require RAM to be allocated to provide sufficient memory space. Memory space is finite, even on an Archimedes, and can be particularly limiting on a 305 (we would suggest all 305 users consider carefully the benefits of upgrading to 1 Mbyte of memory). Even with a 310, and indeed a 440, it is not sensible to allocate the maximum amount of memory for each purpose that you are ever likely to need. For example, animation programs using bank switching in 256 colour modes (see our current series *Animating Archie* for more information) need as much as 160K or even 320K of screen memory. But most applications require far less.

There are two commands which you need for reconfiguring, *STATUS and *CONFIGURE. The former simply displays the current settings of all the parameters which you can change. *CONFIGURE is the command you need to make any changes. All the information required is given in the User Guide in the chapter entitled *Operating System Commands*, but some additional advice may be helpful.

Some settings use two different words to toggle between two settings (rather than alternative parameters with the same word). Thus:

```
*CONFIG. CAPS
or *CONFIG. NOCAPS
```

can be used to determine whether a machine powers up with Caps Lock set or not. I find it easy to forget this, and waste time trying out different parameter settings. Note also the abbreviation for *CONFIGURE.

As delivered, the Archimedes always activates the Desktop after power-up or on pressing Ctrl-Break. Many users prefer their machine to activate Basic. This can be achieved by entering:

```
*CONFIG. Language 4
```

You can, of course, set the Archimedes to default to any module, and you can determine the corresponding number with the *MODULES command.

Several configuration commands allocate memory for a specific purpose, relocatable modules, screen displays, sprite storage and so on. Unfortunately, memory size is specified not in bytes but in so-called pages. Not only that, but page sizes are different for different purposes and on different machines. The situation is summarised in Table 2.

FUNCTION	PAGE SIZE	
	A305/A310/410	A440
Fontsize	4K	4K
RAMFSsize	8K	32K
RMAsize	8K	32K
Screensize	8K	32K
Spriteize	8K	32K
Systemsize	8K	32K

Table 2. Memory page sizes

Thus a screen mode needing 80K of memory (such as mode 12) needs a minimum *screensize* of 10 (that is 10 pages of 8K) on a 305, 310 or 410, but a *screensize* of only 3 (3 pages of 32K = 96K), but no less than this on a 440. If you know which mode a program is using, then the *screensize* settings are easy to calculate. For applications such as sprites and fonts such calculations are less easy, and an intelligent guess maybe a better approach, refining this later.

It is not always necessary to reconfigure memory allocations yourself using *CONFIGURE. For example, if you try to load a relocatable module when Basic is active (indicated by the '>' prompt), you may encounter the message "No room in RMA". Now you can use *CONFIGURE to allocate more pages of memory to *RMA*size and then press Ctrl-Break before trying again. But you may not know how much memory is needed, and either allow too little, and then have to repeat the exercise, or allocate too much and waste the remainder.

The solution is to QUIT from Basic to Arthur (the prompt changes to a '"'). If you then try to load a relocatable module (by prefixing its name with '"') or using the *RMLOAD command), Arthur will automatically allocate the memory needed (though it will not change the status setting).

The reason for this is that loading a relocatable module requires memory that might otherwise be used by Basic. Once Basic is active this cannot be changed. But when we are talking directly to Arthur there is no problem of course. This is very useful if you want to create a boot file (using *BUILD) to load in some relocatable modules before chaining a Basic program. Just make the first command in the boot file QUIT (this gets you out of Basic and into Arthur), followed by commands to load the modules, and then *BASIC before chaining the relevant Basic program. The modules will always load regardless of the *RMA*size setting.

MANIPULATING VARIABLES

A group of star commands provides for values to be assigned to special operating system variables quite separate to any Basic

variables, and for the contents of those OS variables to be displayed. The commands involved (all star commands) are ECHO, SET, SHOW and UNSET. In general, these commands can be entered with user-defined OS variables, or more often, with certain *system variables*.

First of all, let us see how these commands might be used with simple OS variables. For example:

```
*SET WELCOME "Hello World"
*ECHO <WELCOME>
```

would assign the text given to the OS variable WELCOME, while the ECHO command would cause that message to be displayed on the screen (note the angle brackets). If we were to write:

```
*ECHO WELCOME
```

then the string "WELCOME" would be displayed, not the string assigned to the OS variable WELCOME. The command *SHOW will not only show the value currently assigned to an OS variable with *SET, but also show the *type*, here string. Thus:

```
*SHOW WELCOME
```

gives the response:

```
WELCOME : type String, value : Hello
World
```

Arthur also provides for the use of system variables. Some of these are recognised by Arthur, while others may be known to other modules. For example, Sys\$Time, Sys\$Date and Sys\$Year are always active and have the obvious meanings. You could use *SET to change any of these if you so wished, for example:

```
*SET Sys$Year 1989
```

to change the year to 1989 (quotes are unnecessary unless spaces are included in a string).

Another system variable is Cli\$Prompt, which determines the prompt used by Arthur (normally '*'). If you want to change it to '=>' for example, then type:

```
*SET Cli$Prompt =>
```

QUIT Basic and you will see the new prompt in action. Incidentally, when you are communicating directly with Arthur, you don't need to include the '*' with any star commands, though no error arises if you do.

One useful application of system variables is with long pathnames when accessing files. Suppose you had set up a disc with a directory called PD containing a directory called Text containing a directory called Letter. Then:

```
*SET Letters $.PD.Text.Letter
would enable any document in the directory
Letter (say Bank) to be referenced as:
<Letters>.Bank
```

Regardless of the current directory, a correct reference would always be made.

Another very useful variable is used for setting up aliases. An alias is just an alternative name for something. For example, entering:

```
*SET Alias$AID HELP
```

would establish AID as a valid alternative to the operating system command HELP. Then writing *AID would have the same effect as writing *HELP. In fact, Arthur uses this facility to set '.' as an alias for the *CAT command used with the ADFS. Thus *. has the same meaning as *CAT.

Aliases can also be set up to use parameters, and some examples exist by default. For example, this feature is used by Arthur when you attempt to access any file by prefixing the file name with a '*'. Arthur uses predefined aliases to match up the *file type* and substitute the correct command. Thus if you attempted to access a screen display saved as Screen01 by typing simply *Screen01, Arthur would use a pre-defined alias to match the file type and turn the command into *SCREENLOAD Screen01 before passing this to the Command Line Interpreter (the code which first interprets all star commands).

The command *SHOW can be used to show the current setting for any alias, e.g.:

```
*SHOW Alias$AID
```

This command can also use wild card characters. Thus:

```
*SHOW Alias$*
```

will list out all current aliases, while:

```
*SHOW
```

will list all current settings.

We will cover the details of parameter setting in aliases next month (a tricky little subject), and also look at some of the other operating system commands.

HEALTHDATA

The on-line health information database is now available for use off-line on the Archimedes

Some of the most useful and relevant information from Healthdata is now available on disc for the Archimedes. Presented as videotext pages linked by a simple, menu driven structure the Archimedes version has the "look and feel" of accessing a remote database without the disadvantages of high telephone charges.

Topics include:

child health • vaccination • common ailments • vitamins • alcohol • radiation risks • dental treatment travel abroad • women's health • heart disease • contraception • AIDS • diet • sexually transmitted disease • hay fever • pollution • smoking • cystitis • pregnancy • using the NHS • drug abuse • cancer tests • choosing a doctor • back pain • sickle cell disease • thrush • food additives • blood pressure • flu glue sniffing • pre-menstrual syndrome • weight reduction.

Help is included and the carousel feature displays a sequence of index frames. Details of where to go for further information or advice are given at the end of each section. There is a comprehensive index which allows subjects to be found easily. Healthdata is an invaluable reference for home, school or work.

Healthdata will work on all versions of the Archimedes including the A305, A310, A410 and A440. It is supplied on a single 3.5" disc and network versions are also available. Price is just £14.95 or £24.95 for Network version (No VAT). Price includes postage and packing in UK (Overseas orders please add £2.50). Please send cheque payable to Healthdata to: Healthdata, 21 Vicers Close, London, E9 7HT.

MICRO STUDIO

ARCHIMEDES GRAPHIC LIBRARY
THREE DISK SET
£19.95 inc VAT and p&p

BEEB TO ARCHIMEDES DATA LEAD
INCLUDING SOFTWARE
£14.95 inc VAT and p&p

5 1/4 40/80 DISC DRIVE COMPLETE WITH
ARCHIMEDES INTERFACE
£129.95 inc VAT and p&p

*You may pay by Access and Visa,
Educational Orders Welcome.*

MICRO STUDIO(R)
22 Churchgate St., Soham,
Nr Ely, Cambs, CB7 5HL.

DESKTOP DIARY UTILITY

Make your

"Archimedes Desktop Diaries"
convenient to use.

See a full year at a glance.

A single screen shows part of the entry for every day of the year. See the free days or spot that elusive entry at a glance.

Instant viewing of the full entry for any day.

Skim the mouse over the year to scan the full entries for any day, & just 'click' on a day to make an entry.

Print all of a month with just one command.

Select the month, or months, to be printed. Then choose from either a full page, weekly formatted print-out, or a quick single line per day, showing only those days which have an entry.

Easy to use, includes Help & *commands.

£9.95

(incl. vat & U.K. p&p)
Official Orders also welcome.

Cheque or P.O. to

QUERCUS COMPUTER SYSTEMS
Courtyard House, North Otterington,
N.Yorks. DL7 9EP

QUERCUS specialises in customised technical & process control software, on a wide variety of micros. Tel. 0659 70643 to discuss your requirements.



MOUSE MENU

David Spencer describes a universal WIMP driven menu.

Anyone who has used the RISC User monthly disc will be familiar with the menu that is included each month. For those who haven't seen the disc, the menu lists all the items available. By moving the pointer and clicking on any item, a box containing further information on that option pops up. Double clicking on an item will perform the task associated with that item; normally this means that the appropriate program is loaded and run.

The program given here is designed to offer a broadly similar menu facility, but one which is fully WIMP based (unlike that on the magazine disc), and which can be readily incorporated into your own programs, either as part of an existing WIMP program, or by inserting some initialisation routines, in a non-WIMP program.

The listing given here consists of the routines that make up the menu system, along with a sample program that displays a typical menu. If you type in, save and then run the program, you will be presented with a three entry menu. The first and last entries consist of a title followed by some brief explanatory text, while the second menu has just a title.

Clicking the select button on either of the first two entries will bring up an information box containing more details of that option. This does not occur for the third entry which has no additional information.

Double clicking on a particular option will perform a task related to that menu option. For the first two entries this merely causes some text to be printed at the top left of the screen, while the third entry plays a musical note and quits the menu.

USING THE MENU SYSTEM

To use the menu system in your own programs, you will need to incorporate the procedure `PROCinit` and the procedure `PROCmenu`, and all those routines which follow it. Calling `PROCinit` sets up the WIMP environment for the menu. If your program already uses windows then much of this routine

MULTI-TASKING WITH RISC OS

This program is fully compatible with RISC OS, but by deleting lines 130, 180 and 200-390 it can be made to operate in full multi-tasking mode.

could be omitted, though it will still be necessary to dimension the arrays.

A menu is called up using:

`PROCmenu (number, Xpos, Ypos)`

where *number* is the menu number, and *Xpos* and *Ypos* are the co-ordinates at which the top left-hand corner of the menu should appear. There is no reason why a program should not have more than one WIMP menu if required.

CREATING YOUR OWN MENUS

The information that forms each menu is contained within `DATA` statements which may be located anywhere in the program. The first line of menu data must be of the form:

`.menu<n>`

where *n* is the menu number, for example, `.menu1`, and this is followed by the menu title. It is this title that appears at the top of the menu window.

The menu entries then follow in order, with the end of the menu being marked by a `DATA` statement containing the text `'!MENUEND'`. You can see the pattern by looking at the `DATA` statements at the end of the example program. Each menu entry consists of:

A numeric *run type*.

A command string, the function of which depends on the *run type*.

An entry heading.

Any number of lines of subsidiary text that are printed indented under the heading.

The line `'!TEXT'`.

The lines of text that make up the information box for that entry (see below for more details).

The line `'!END'`.

The *run type* is a number which determines the action to be taken when the select button is double-clicked over that option. The current options are:

MOUSE MENU



1. Print the command string in the top left-hand corner of the screen (this is only included as a demo).
2. Execute the command string as a star command.
3. Chain the Basic program using the command string as a filename.

More *run* types could be added by extending the options in the CASE statement between lines 2370 and 2420. If the *run* type given in the menu entry is negative, then the menu will be quit once the option has been executed. Obviously, loading in a new program will exit from the menu anyway.

The entry heading, and any subsidiary text, are limited in length to forty characters per line. On the other hand, the information box will be made the correct width for the longest line of text, up to a limit of 78 characters. Additionally, because the information box cannot scroll vertically, there is a maximum limit of about twenty-five lines.

Studying the example menu from the program should clarify the above description of the menu format.

To illustrate further the use of this program, this month's magazine disc contains a sample menu that can be used to select and run any program from that disc.

```

10 REM >WimpMenu
20 REM Program WIMP Based Menu
30 REM Version A1.0
40 REM Authors David Spencer
50 REM and Felix Andrew
60 REM RISC User December 1988
70 REM Program Subject To Copyright
80 :
90 PROCinit
100 ON ERROR GOTO 120
110 PROCmenu(1,500,800)
120 SYS"Wimp_CloseDown"
130 MODE0:*FX4
140 END
150 :
160 DEF PROCinit
170 DIM block 100,start(20)
180 MODE 12
190 $block="TASK":SYS "Wimp_Initialise

```

```

",200,!block,"Universal Menu"
200 COLOUR 0,&F0,&F0,&F0
210 COLOUR 1,&D0,&D0,&D0
220 COLOUR 2,&B0,&B0,&B0
230 COLOUR 3,&90,&90,&90
240 COLOUR 4,&70,&70,&70
250 COLOUR 5,&50,&50,&50
260 COLOUR 6,&30,&30,&30
270 COLOUR 7,0,0,0
280 COLOUR 8,0,&40,&90
290 COLOUR 9,&E0,&E0,0
300 COLOUR 10,0,&C0,0
310 COLOUR 11,&D0,0,0
320 COLOUR 12,&E0,&E0,&B0
330 COLOUR 13,&50,&80,0
340 COLOUR 14,&F0,&B0,0
350 COLOUR 15,&70,&70,&70
360 VDU 19,0,24,&70,&70,&70
370 VDU 19,1,25,0,&F0,&F0
380 VDU 19,2,25,0,0,&90
390 VDU 19,3,25,&F0,0,0
400 ENDPROC
410 :
420 DEF PROCmenu(num,X,Y)
430 LOCAL reason,depth,title$,data$,po
pped,quit,menuw,infow
440 PROCgetdepth
450 PROCcreate
460 REPEAT
470 SYS "Wimp_Poll",,block TO reason
480 WHILE reason
490 CASE reason OF
500 WHEN 1:PROCredraw(!block)
510 WHEN 2:SYS"Wimp_OpenWindow",,block
520 WHEN 3:PROCclose(!block)
530 WHEN 6:PROCbuttons(block!8)
540 WHEN 17,18:IF block!16=0 THEN quit
=TRUE
550 ENDCASE
560 SYS "Wimp_Poll",,block TO reason
570 ENDWHILE
580 UNTIL quit
590 PROCdelete(menuw)
600 PROCdelete(infow)
610 ENDPROC
620 :
630 DEF PROCdelete(wind)
640 !block=wind
650 SYS "Wimp_DeleteWindow",,block
660 ENDPROC
670 :
680 DEF PROCgetdepth
690 PROCrestore(num)
700 depth=-1

```




MOUSE MENU

```
710 READ title$
720 REPEAT
730 READ data$
740 IF data$<>"!MENUEND" THEN
750 READ data$
760 REPEAT
770 READ data$:depth+=1
780 UNTIL LEFT$(data$,1)="!"
790 IF data$<>"!END" THEN
800 REPEAT:READ data$:UNTIL data$="!EN
D"
810 ENDIF
820 ENDIF
830 UNTIL data$="!MENUEND"
840 ENDPROC
850 :
860 DEF PROCcreate
870 menuw=FNcreate(X,Y,656,depth*32+32
,7,2,title$,7,0,3,1,7)
880 infow=FNcreate(0,0,0,0,7,14,"Infor
mation",7,0,3,1,&83)
890 PROCopen(menuw,X,Y,656,depth*32+32
)
900 popped=FALSE
910 quit=FALSE
920 ENDPROC
930 :
940 DEF PROCclose(wind)
950 !block=wind
960 SYS "Wimp_CloseWindow",,block
970 IF wind=menuw THEN
980 quit=TRUE
990 IF popped THEN
1000 !block=infow
1010 SYS "Wimp_CloseWindow",,block
1020 ENDIF
1030 ENDIF
1040 ENDPROC
1050 :
1060 DEF PROCrestore(label)
1070 LOCAL A$:RESTORE
1080 REPEAT READ A$
1090 UNTIL A$=".menu"+STR$label
1100 ENDPROC
1110 :
1120 DEF PROCopen(handle,x,y,w,d)
1130 !block=handle
1140 block!4=x:block!8=y-d
1150 block!12=x+w:block!16=y
1160 block!20=0:block!24=0
1170 block!28=-1
1180 SYS "Wimp_OpenWindow",,block
1190 ENDPROC
1200 :
1210 DEF FNcreate(px,py,pw,pd,tf,tb,T$,
pf,pb,sf,sb,tflags)
1220 LOCAL handle
1230 $block=STRING$(92,CHR$0)
1240 block!0=px:block!8=px+pw
1250 block!4=py:block!12=py-pd
1260 block!24=-1:block!28=tflags
1270 block?32=tf:block?33=tb
1280 block?34=pf:block?35=pb
1290 block?36=sf:block?37=sb
1300 block!44=-pd:block!48=pw
1310 block!56=&3D:block!60=&A000
1320 T$=LEFT$(T$,11):$(block+72)=T$
1330 SYS "Wimp_CreateWindow",,block TO
handle
1340 =handle
1350 :
1360 DEF PROCredraw(handle)
1370 LOCAL more
1380 SYS "Wimp_RedrawWindow",,block TO
more
1390 WHILE more
1400 IF !block=menuw PROCredrawmenu ELS
E PROCredrawinfo
1410 SYS "Wimp_GetRectangle",,block TO
more
1420 ENDWHILE
1430 ENDPROC
1440 :
1450 DEF PROCredrawmenu
1460 LOCAL x,y,line,item
1470 x=block!4-block!20:y=block!16-bloc
k!24-16
1480 PROCrestore(num):READ data$
1490 line=0:item=0
1500 READ data$
1510 WHILE data$<>"!MENUEND"
1520 READ data$
1530 READ data$:PROCprint(data$)
1540 start(item)=line:item+=1
1550 READ data$
1560 WHILE LEFT$(data$,1)<>"!"
1570 PROCprint(" "+data$)
1580 READ data$
1590 ENDWHILE
1600 IF data$<>"!END" THEN
1610 REPEAT
1620 READ data$
1630 UNTIL data$="!END"
1640 ENDIF
1650 READ data$
1660 PROCprint("")
1670 ENDWHILE
1680 start(item)=line+1
```

MOUSE MENU



```

1690 ENDPROC
1700 :
1710 DEF PROCredrawinfo
1720 LOCAL x,y
1730 x=block!4:y=block!16-16
1740 PROCgoto(popped)
1750 REPEAT
1760 READ data$
1770 UNTIL data$="!TEXT"
1780 REPEAT
1790 READ data$
1800 IF data$<>"!END" THEN PROCprint(da
ta$)
1810 UNTIL data$="!END"
1820 ENDPROC
1830 :
1840 DEF PROCprint(line$)
1850 MOVE x+8,y
1860 PRINT line$
1870 y:=32;line+=1
1880 ENDPROC
1890 :
1900 DEF PROCbuttons(pressed)
1910 LOCAL my
1920 IF block!12=infow THEN
1930 SYS "Wimp_CloseWindow",,block+12
1940 popped=FALSE
1950 ELSE
1960 IF block!12=menuw THEN
1970 SYS "Wimp_GetWindowState",,block+1
2
1980 my=block!4-block!28+block!36
1990 IF pressed=1024 THEN PROCpopup(my,
block!16,block!4)
2000 IF pressed=4 THEN PROCchosen(my)
2010 ENDIF
2020 ENDIF
2030 ENDPROC
2040 :
2050 DEF PROCpopup(ypos,X,Y)
2060 LOCAL lines,len,item
2070 PROCcloseinfo
2080 item=FNwhich(ypos)
2090 IF item THEN
2100 PROCgoto(item)
2110 REPEAT
2120 READ data$
2130 UNTIL LEFT$(data$,1)="!"
2140 lines=-1:len=0
2150 IF data$="!TEXT" THEN
2160 REPEAT
2170 READ data$:lines+=1
2180 IF data$<>"!END" AND LENdata$>len
THEN len=LENdata$
2190 UNTIL data$="!END"
2200 !block=0:block!8=(len+1)*16
2210 block!4=-(lines+1)*32:block!12=0
2220 SYS "Wimp_SetExtent",infow,block
2230 PROCopen(infow,X+400,Y,(len+1)*16,
(lines+1)*32)
2240 popped=item
2250 ENDIF
2260 ENDIF
2270 ENDPROC
2280 :
2290 DEF PROCchosen(ypos)
2300 LOCAL item,runtime
2310 PROCcloseinfo
2320 item=FNwhich(ypos)
2330 IF item THEN
2340 PROCgoto(item)
2350 READ runtime,data$
2360 IF runtime<0 THEN quit=TRUE:PROCcl
ose(menuw)
2370 CASE ABSruntime OF
2380 WHEN 1:VDU 4:PRINT TAB(0,0);data$;
SPC4:VDU5
2390 WHEN 2:OSCLI data$
2400 WHEN 3:CHAIN data$
2410 REM Insert other types here
2420 OTHERWISE
2430 ENDCASE
2440 ELSE
2450 VDU 7
2460 ENDIF
2470 ENDPROC
2480 :
2490 DEF PROCcloseinfo
2500 IF popped THEN
2510 !block=infow
2520 SYS "Wimp_CloseWindow",,block
2530 popped=FALSE
2540 ENDIF
2550 ENDPROC
2560 :
2570 DEF PROCgoto(item)
2580 LOCAL count
2590 PROCrestore(num)
2600 READ data$
2610 IF item<>1 THEN
2620 FOR count=1 TO item-1
2630 REPEAT
2640 READ data$
2650 UNTIL data$="!END"
2660 NEXT
2670 ENDIF
2680 ENDPROC

```

Continued on page 36



Euclid

The 3-D modelling
and animation system for
the Acorn Archimedes.



Here's what the reviewers say:

“...remarkable value. It already matches or exceeds many functions of other visualisation packages—most of which are far more expensive.”

A&B Computing October 1988

“...very useful indeed.”

Micro User September 1988

“...Euclid will be a great success and a useful tool for any future 3D design programs (a standard even).”

Archive July 1988

“...well worth the attention of anyone interested in this field.”

RISC User July/August 1988

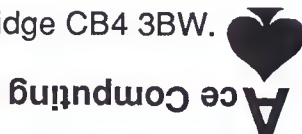
EUCLID - Explore a new dimension!

Price: £45 (inc VAT and P&P).

Available by mail order from:

Ace Computing, 27 Victoria Road, Cambridge CB4 3BW.

Or from your local dealer.



Archimedes Extravaganza

ABC: The Archimedes Basic Compiler

"ABC is a vital part of any programmer's toolbox, it puts compilers on other systems to shame. Unquestionably one of the most impressive pieces of software I have yet seen running on the Archimedes." A&B Dec 1988

As A&B Computing found, the Archimedes Basic Compiler makes writing machine code programs and relocatable modules as easy as ABC! Programs can be written using the full range of BASIC V's error messages and report's, and once fully working, run through ABC which transforms them into machine code, ready for immediate action. ABC accepts assembly listings within programs.

ABC supports a comprehensive range of Archimedes Directives to enable you to control how the compiler works to suit your own requirements. Programs may be compiled either in RAM or to and from disc. The code produced is totally stand alone (the FPE may be needed). ABC is extremely easy to use and speed increases of up to and over 4000% are possible.

Demo Disc: We have produced a demo disc of ABC which still supports over 100 commands. It costs just £2 and this is refundable on any subsequent purchase. A full specification sheet is also available on request.

ABC was written by Paul Fellows head of the team that wrote the Archimedes Operating System. Is there a better qualification?

"I was impressed...The code is efficient...and the speed is justifiably fast...the compiler is very easy to use..." A&B.

£99.95

Arcendium: Board Game Fun for All!

Backgammon, Draughts, Reversi and Quadline – four games for the price of one! Sure to get the whole family using the Archimedes.

Arcendium allows numerous levels of play either against an opponent or the computer. Sound, speech and superb colour graphics all add to the fun. Includes on-line help and re-play facilities plus more...

£14.95

Alerion: Archimedes Arcade Action!

The highly acclaimed all-action shoot-em-up for the Archimedes. 256 colour mode graphics, with digitised speech. Impossible to finish! Superb fun!

"...the gameplay makes this game a winner...a first rate game..." A&B Computing.

£14.95

Archimedes PC Emulator Shareware

Five discs full of PC software tested with the emulator to ensure compatibility. The collection includes software you would normally expect to pay a fortune for and includes a word processor, spreadsheet, games, flowchart designer, printer utilities, and very much more.

Mindreader is a powerful word processor with mailmerge, spelling checker, and an amazing 'word anticipate' feature. As Easy As is a full-function Lotus-compatible spreadsheet, with macros and graphics. Games include Bridge, colour chess, plane-flying etc. Utilities include a writing style analyser, font designer, letter-quality text printer ... and so the list goes on. Nearly 4Mbs of PC action complete with User Guide.

£34.95

Dabhand Guides for your Archimedes

In reviewing our books PCW said this: *"...a truly professional publication."*

C: A Dabhand Guide

PCW said: *"I only wish this book had been available when I was learning C."* If you want to learn C on your Archimedes then this 512 page volume is the way to do it.

At £14.95 it represents quite incredible value. Book and programs disc £21.95. Excellent value.

Archimedes Assembly Language

386 pages devoted to programming the Archimedes in machine code. At £14.95 it is the only book which deals specifically with assembler on the Archimedes. Book and programs disc £21.95.

Coming Soon: Jan 89

Archimedes Operating System: Our book on the Operating System and how to use it. Available end of January 1989.

BASIC V by Mike Willaims – all you want to know about BBC BASIC on your Archimedes for £9.95. Available January 1989.

Instigator – the RISC OS compatible Archimedes System Manager. Over 65 invaluable * commands in a Relocatable Module. £49.95.

Alped the ultimate graphics adventure by Felix Andrew. £14.95

**FREE on Request
Bumper Catalogue!**

DABS PRESS

5 Victoria Lane (RUJ), Whitefield, Manchester M25 6AL

Tel: 061-766 8423 BT Gold: 72:MAG11596 Prestel: 942876210

Prices include VAT and P&P (UK/BFPO/CI). ACCESS/VISA accepted by post/phone/Mailbox/ in person. Cheques and POs to address above. Dabs Press products available from all good dealers. Add £2.50 (£12 air) if outside UK. Official orders welcome.



THE ABC BASIC COMPILER

Reviewed by Mark Sealey

Dabs Press is devoting more and more time to the Archimedes these days, and has already produced books on C and ARM Assembler. Dabs Press has now reinforced its market position with a fully fledged Basic compiler, called ABC (Archimedes Basic Compiler), a significant piece of software with implications for both software developers and consumers alike.

If a Basic program can be converted into machine code, it will almost always run much more quickly. Dabs Press claims that improvements of up to 40 times in some instances can be achieved with ABC. Even a simple FOR-NEXT loop (of some 7000 iterations), amongst programs tested for this review, was almost 1000% faster. There is, however, likely to be a wide variation in speed gain, and some routines may actually be slower. Comparative execution times for the standard PCW benchmarks are shown in Table 1 overleaf.

ARCHIMEDES BASIC COMPILER

Such an increase in speed is just what ABC is designed to achieve, and it does this very well. The bulk of the software is contained in a relocatable module, which itself needs 200K of memory. Once the (source) Basic program is on disc, it is only necessary to run the compiler to produce equivalent machine code.

This object code is never less than 16K in length regardless of the size of the source program, and is always much larger than the original. For example, a 1K Basic program compiled to 17K of machine code, a 2K program compiled to 19K, a 9K program to 28K and a 22K program to 76K. The compiler takes from 10 seconds (on the shortest of programs) up to 5 minutes on a 22K Basic program.

The object code file may be saved to disc, and then executed without any further reference to ABC. However, compiled programs depend upon the floating point emulator for all real number arithmetic, and this leads to incompatibilities compared with

INTERPRETERS VERSUS COMPILERS

Any computer acts directly on a binary sequence of digits in the form of machine code, but most programs are written in high level languages such as Pascal and Basic, which have instructions nearer to normal English. The gap between high level language and machine code is bridged with either an interpreter or a compiler.

Basic, for example, is normally an *interpreted* language. Each Basic instruction to be executed is analysed by the Basic interpreter, which then carries out the functions specified in that statement by reference to its own machine code routines. For example, if a Basic program contained an instruction to multiply together the values of two variables, the Basic interpreter, having determined this first, would extract the two values concerned and pass them to its own multiply routine to obtain the result.

The advantages of this approach are that the original Basic program is always present and can be listed, altered and tested at will. But during execution, the frequent analysis (interpretation) of every statement, particularly when it is in a loop, is time consuming. Even on an Archimedes, pure machine code can be much faster, sometimes significantly so.

the way in which real numbers are handled by Basic's own floating point routines. These differences are described in the manual, with hints on how any problems may be avoided.

The compiler also supports the full ARM assembler instruction set, which copes with any assembler code embedded in a Basic program. Where it occurs, the compiled program still assembles its own code at run-time, and that still requires two passes. Again, the manuals have an instructive chapter on ARM assembler thrown in for good measure.

RUNNING ABC

Once the compiler module has been loaded, it can be called with the command *COMPILE. This can take two optional parameters, the file names of the source Basic program and the object code file to be created. For example:

THE ABC BASIC COMPILER

*COMPILE

or:

*COMPILE MYPROG OBJECT

Benchmark	Basic V	ABC	ratio	increase %
GRAFSCRN	1.68	0.84	2:1	100
STORE	6.50	6.50	1:1	0
TEXTSCRN	2.46	2.33	1:1	0
INTMATH	0.92	0.19	9:1	800
REALMATH	0.26	0.30	1:1	0
TRIGLOG	1.20	3.38	1:3	-200
SIEVE	5.16	0.58	9:1	800
Recursive Functions				
TACK	27.53	0.78	25:1	2400
FIBONACCI	49.43	1.40	35:1	3400
ACKERMAN	4.89	0.12	40:1	3900
Array Operations				
INT ARRAY	1.84	0.34	5:1	400
REAL ARRAY	2.04	1.34	1.5:1	50
STRING ARRAY	2.40	1.60	1.5:1	50

**Table 1. PCW Benchmarks provided by
Dabs Press**
(execution times in seconds)

ABC will present you with a simple, uncluttered four-part panel. One part allows you to input source and object file names, if not already entered, and set one option (of which more presently). The second part of the panel reflects what is happening, and prompts for action should a compile-time error occur. In the middle of the display is a real-time, horizontal bar graph indicating progress through the four-pass compilation process. The fourth and largest window can be used to display the source listing and any star commands (using Alt-*).

When an error does occur during compilation, you are offered the opportunity of dropping directly into the ARM Basic Editor to doctor the offending line. Unfortunately, you cannot simply return to ABC, but must first re-save the edited program before re-issuing the *COMPILE command. Just what these errors might be and how to correct them is dealt with well in the manuals. For instance, ABC will not support procedures with multiple exits (more than one ENDPROC per procedure definition).

Once any fatal error has been detected, the problem must be corrected before attempting re-compilation. Warnings, on the other hand, can be ignored if you choose, allowing compilation to continue. Users should also be aware that setting certain error conditions in the Basic source program can send the compiler into an infinite loop, which can only be resolved with Ctrl-Break.

Any compiler must analyse all the instructions in a program, and so every statement is checked syntactically, even if some of the code is unlikely to be executed except in the most unusual circumstances. This usefully uncovers potential trouble spots which might otherwise remain undiscovered until too late (certainly some of the supposedly well-tried test programs used with the compiler revealed the odd problem - uninitialised variables for example).

DIRECTIVES

ABC also allows any of some seven *directives* to be inserted into a Basic program before compilation. These take the form of special REM statements with a parameter contained in curly brackets. For example, the directives *REM {LIST}* and *REM {NOLIST}* may be used to determine which sections of a Basic program will be listed on the screen during compilation. Foregoing the listing of source code at this time does reduce the compilation time (for example, from 317 seconds to 277 seconds for a 22K Basic program), although the differences are not that significant.

The same technique, using *REM {NOCOMPILE}* and then *REM {COMPILE}*, represents an even greater time saving since you can test-compile further and further into a long program, restarting compilation at a progressively later point after removing earlier errors. When debugging is finished, you could remove such directives altogether.

The most important of these directives, though, concerns memory management. The ABC Reference Guide explains how the Archimedes' RAM is utilised during compilation for stack and heap operations. ABC chooses

default values for these, but the user can specify directives to select alternatives. During compilation, ABC continually checks the current stack and heap limits for any conflict. This is quite time consuming and can be omitted by inserting `REM {NOSTACKCHECK}`, to suppress checking. This is also useful for long programs, where the presence in memory of ABC, the Basic source program and the generated object code might well result in "stack overflow" errors which would not arise were the compiled machine code to be executed on its own.

Another way around this, is to load and save source and/or object code directly from disc, instead of trying to hold both together in RAM (the normal default). This can be specified as an option at the start, by changing the default setting of 'RAM' to 'DISC'. Sometimes - with a file of over 50K for example - it is necessary to take both steps (even on a 1 Mbyte machine): compile from disc and set `{NOSTACKCHECK}`. All files tried were eventually compiled while testing ABC. There is no facility for compiling and linking separate modules.

RESTRICTIONS ON BASIC V

There are, in addition, other restrictions. The `EVAL` function (which evaluates expressions at run time) cannot be compiled for just that reason. To compensate for this, however, ABC uses a slightly enhanced and standardised input format to allow, for instance, discrimination between hex, binary or decimal digits at all times. The ABC Reference Guide also has a very useful and comprehensive survey of all Basic V keywords as treated by the compiler. This helps in making any slight changes to the Basic code necessary to enable it to compile, (ten out of a random selection of 20 programs tried compiled first time).

Other Basic keywords not allowed or not implemented include `APPEND`, `CHAIN`, `CLEAR`, `LOAD` and `SAVE`, `COUNT`, `SUM` and `WIDTH`, `TWIN`, `EDIT`, `INSTALL` and the like. The manual does explain why these are not supportable, and gives some ways round the

limitations. Nor does ABC at present support the extended array functions of Basic V, local array handling, or the passing of arrays as parameters. Assignment of pseudo-variables (`HIMEM`, `PAGE` etc) is not possible either, which might create some difficulty in programs which use them. But for everyone who finds this really restrictive, there will be many more glad of the final major feature of this excellent Dabs product.

COMPILING RELOCATABLE MODULES

To turn your program into a relocatable module, it is only necessary to include at the head of your Basic program five or six directives such as:

```
20 REM {MODULE TYPE SERVICE}
30 REM {MODULE TITLE BAUDSET}
40 REM {MODULE VERSION 1.04}
```

Although ABC does not, at present, allow modules which take more than a default parameter (e.g. `*BUFFER` as opposed to `*BUFFER 2048` or `*BUFFER 1024`), this is planned for the next release of ABC.

VERDICT

Buy it! Despite some inevitable limitations on what may be compiled, ABC is quite comprehensive, and many programs tried either compiled immediately or after a few fairly obvious changes. The one feature of Basic V which cannot be got round is the impossibility of compiling the `EVAL` function. The compiler is simple and straightforward to use, and the User Guide and Reference Guide helpful and instructive. Communicating with ABC is not always as smooth as one might wish, but that is generally a minor failing in an otherwise excellent product. Version 2 of ABC, which removes many of the limitations described above, will be available very shortly and will be supplied free of charge to all registered users.

I had hoped to compare ABC with a rival offering from Silicon Vision, but this was not available in time. We will review the Silicon Vision compiler in a future issue of the magazine.

ABC, Archimedes BASIC Compiler (£99.95)
Is supplied by DABS Press, Victoria Lane,
Whitefield, Manchester M25 6AL.
Tel. 061-766 8423

Leonardo

Archimedes Mode 12 Art Package

A comprehensive set of drawing options all available at four levels of magnification

Quick compressed screen files, so you can save many more screens on a disk

Available for all models of Archimedes

£17.50 inclusive

Leonardo 256

Archimedes Mode 15 Art Package

Mode 15 version of Leonardo with 256 colours plus a font generator

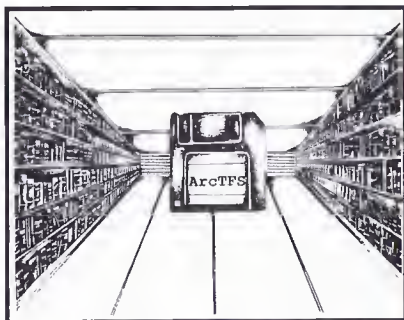
(Leonardo 256 requires 1 megabyte of memory)

£19.50 inclusive

Available from : Beard Technology
111 Evering Road
London, N16 7SL.

ArcTFS - THE RESEARCH & AUTHOR'S DATABASE

- * Indexes, cross-references and files typed or imported ASCII text items up to 7700 characters in length.
- * Keeps searchable alphasorted files of explainers for all indexing and source codes.
- * Creates additional referencing fields to suit the subject.
- * Creates text files up to 650K in length on a 800K disc - the equivalent of 80,000-100,000 words.
- * Searches text items directly or any referencing field.
- * Formats any subset and saves it as a file which will be read as a default - formatted native file, answerable to all reformatting commands, by ArcWriter or 1st Word Plus.
- * Translates Wordwise or View files in ASCII form into ArcWriter or 1st Word Plus format, and gives database indexing and control of these files in the meantime.



"There is no doubt about the value of ArcTFS to the writer or the researcher. It's easy to use, and allows sophisticated searching and sorting of text. Anyone who needs to manage large amounts of this sort of information should find ArcTFS invaluable" Dave Fletcher Acorn User November 1988.

ArcTFS (Text Filing System) - £29.95

(Cheque or PO only) to:

TEXCELLENCE, 2 Greenhill Road, Coleraine, N.Ireland BT51 3JE
(For further details send SAE)



MOVIE MAKER

by Lee Calcraft

Create your own cartoons and animated sequences with this full-feature application. This incorporates part 5 of Animating Archie.

This program needs a screen size of 160K

Movie Maker allows the creation of a wide variety of animated sequences. These may be replayed from within the program, or examined in closer detail using the single-stepping option. Sequences may then be saved in compressed form for later use. The program has many applications, and may be used for creating animated title sequences, short cartoons, moving demonstrations, or just experimenting with animation techniques. Moreover, with the aid of a short additional program, sequences can be played back from within other programs, in such a way as to overlay screens already generated by the user's program. This facility makes Movie Maker ideal for producing animated logos etc.

The program, which works in mode 13, is fully mouse-driven, though to keep the code to a reasonable length, it does not use the Arc's WIMP manager. To get the program running, you should first type it in, and save it to disc. Next you will need a copy of a short piece of ARM code from last month's Animation article. You can obtain this by following the instructions in table 1. Once you have the new "DeltaCode" file on your disc, you can run Movie Maker.

1. Load last month's DeltaFile program.
2. Enter the following 4 lines:
120 GOTO 560 (lines 130-550 not reqd)
1270 MOV scrnpnt, #27*256
1390 CMP scrnpnt, #224*256
1585 STR R0, data2+12
3. Run the program.
4. Follow the screen prompt.

Table 1. Generating the DeltaCode

When you run Movie Maker you will be presented with a screen with two main areas. The upper part contains the drawing area, while below this you will see the controls. There is also an important status readout at the very top



of the screen displaying the number of the current frame, the total number of frames in RAM, the amount of free RAM, and the number of bytes so far used for stored frames. This latter will start at 12 bytes, which is the size of the file's header.

To use the program, simply press *select* when in the main drawing area. This draws on the screen in a similar way to an ordinary painting package. To change the brush, click on *brush*. There are 8 brush types, and the type number is displayed. The left mouse button increases the brush size, while the middle button reduces it. Brushes 0 to 4 give points and blobs, while 5 to 7 are air-sprays. To change the colour, just click on the colour palette. Alternatively, clicking with the middle button while in the drawing area will pick up the colour under the pointer. When you have drawn all you wish to for your first frame, press the right-hand button with the pointer in the drawing area. You will hear a short beep, and the frame will be stored in RAM. The status line changes to reflect this, and you will see that the current frame number is now 2, and the total number of frames stored is 1.

You can continue in this way until you run out of RAM. To see the effect of your creations, click on *Replay*. You will see the frames



replayed in an animated sequence. If you use the middle button on the *Replay* icon, the sequence will be repeated until you click the mouse again. The box immediately below *Replay* displays the current playback frame rate (default 25 frames per sec). To alter this, use the left or middle button of the mouse. Now that you have some frames in RAM you can also make use of the box at the top left of the selection area. This has three functions, depending on the exact pointer position. When the pointer is over the "<<", you will be taken to the start of the sequence, while clicking on ">>" takes you to the end. Clicking on the ">" symbol allows you to single-step through the frames from the current point. This can be quite useful, because it permits you to discard later frames by replacing them with others. For example, if you have a sequence of 100 frames, and feel that the last 10 have gone wrong, just go to frame one, then single-step until you come to the last "good" frame. Then continue from there, drawing new frames, and saving them as before.

As you store each frame, you will see how much RAM it uses, and providing that not too many pixels change between each frame, very little RAM will be used. This is because the program uses the Delta File technique to save only the *difference* between the current frame and the previous one, as explained last month. As a result, you can create relatively long animated sequences - perhaps up to half a minute or so on a 1 M byte Arc at 25 frames per second (more at lower frame rates); and quite a lot can be achieved in half a minute. Of course, how many frames you can squeeze in depends on what they contain.

Movie Maker also boasts a primitive text facility. If you type from the keyboard, the letters will appear on the screen in the current foreground colour. The pointer moves as you type, and if you move the pointer by hand while typing, you can produce a variety of effects. If you make a mistake, the *Delete* key can be used. Alternatively you could select a large brush in the background colour, and paint over the offending areas.

There remain three options not yet mentioned. The *Load* and *Save* boxes allow you to load and save frame sequences, prompting for filename, and so-on. If none is given, the current filename will be used. Note also that clicking on the asterisk on the *Load* box gives access to star commands (press Return to exit). Finally, the *Clear* option (which requires confirmation) clears the screen to the current colour, and resets the animator to the first frame, clearing all contents. The real purpose of this is to allow you to select a starting background colour for a sequence, without massive RAM overheads. The start colour is saved in the frame file, and is used whenever the sequence is shown. If you try to colour a screen by painting it over, you will see that a lot of RAM is used, since the frame difference for the newly-coloured frame will be considerable.

Next month we will add further options to this program, including a Copy feature, which simplifies animation by permitting the user to copy areas of the screen from one place to another.

The magazine disc contains both the necessary Delta Code, and example animations.

```

10 REM                                     >Animator
20 REM Program      Mouse Animator
30 REM Version      A 0.9Gm
40 REM Author       Lee Calcraft
50 REM RISC User    December 1988
60 REM Copyright    Lee Calcraft
70 :
80 MODE15:MODE13:OFF
90 DIM buff $30,code $200
100 bsize%=HIMEM-TOP-65000
110 DIM screens bsize%
120 OSCLI("LOAD DeltaCode "+STR$~code)
130 :
140 PROCscreensetup(TRUE)
150 ON ERROR PROCerror
160 PROCmouse
170 END
180 :
190 DATA << > >>,Replay,Brush,,Load *
```



MOVIE MAKER

```
200 DATA Clear,Fr/s, , , " Save"
210 :
220 DEFPROCscreensetup(all)
230 IF all THEN
240 *POINTER
250 VDU 19,0,24,96,96,124
260 PROCplinth(0,0,1280,1024,16,63+19
2,62+192,57+192,52,32,TRUE)
270 PROCplinth(50,296,1188,648,12,63+
192,62+192,0,52,0,FALSE)
280 A%=screens+bsize%-610
290 CALL code :REM initialise
300 CALL code+4 :REM copy to shadow
310 A%=screens+12
320 frameno=1:frametot=0
330 col=63:tint=192:spr=FALSE
340 startcol=0:starttint=0
350 brush=0:file$="None Known"
360 speed%=2:REM 25 Fps
370 ENDIF
380 COLOUR 128+57 TINT 192:COLOUR 0 TI
NT 0
390 basex=56:basesy=40
400 PROCplinth(basex-12,basesy-12,35*32
+24+40,128+24+100,12,63+192,62+192,57+19
2,52,0,TRUE)
410 PROCdrawgrid(basex,basesy,7*32+8,2*
32,5,2,4,8,57,0)
420 PROCputcols
430 PROCspeed(FALSE)
440 PROCshowbox(col,tint)
450 PRINTTAB(22,27);brush
460 VDU 24,basex+4;300+8;basex+1170;30
0+634;
470 ENDPROC
480 :
490 DEFPROCputcols
500 cpx=76:cpy=180
510 FOR C%=0 TO 255
520 GCOL C% DIV 4 TINT C% MOD 4<<6
530 RECTANGLE FILL cpx+C% MOD 64*16,cp
y+C% DIV 64*20,12,16
540 NEXT:ENDPROC
550 :
560 DEFPROCmouse
570 COLOUR 0 TINT 0
580 COLOUR 57+128 TINT 240
590 REPEAT
600 PROCheader
610 PROCmousewait(-2)
620 IF z%>0 THEN
630 REM IF z%=4 THEN xx%=x%:yy%=y%
640 CASE TRUE OF
650 WHEN FNpositiona(x%,y%,basex+4,
300+8,basesy+1170,300+634):PROCTopmouse
660 WHEN FNpositiond(x%,y%,cpx,cpy,
1024,80):SOUND 1,-10,70,1:PROCcolours
670 WHEN FNpositiond(x%,y%,basex,ba
sey,1160,128):PROCselect
680 ENDCASE
690 ELSE
700 VDU5
710 IF w%>31 AND w%<128 THEN
720 MOVE x%,y%+32:PRINTCHR$(w%)
730 x%+=32+64*(w%=127)
740 MOUSE TO x%,y%
750 ENDIF
760 VDU4:OFF
770 ENDIF
780 UNTIL FALSE
790 ENDPROC
800 :
810 DEFPROCheader
820 PRINTTAB(2,1)"FRAME: ";frameno;" (
";frametot;") "
830 used$=STR$(A%-screens):ul=6-LEN(us
ed$)
840 PRINTTAB(19,1)"FREE: ";(screens+bs
ize%-A%)DIV 1024;"K ";TAB(32,1);SPC(ul)
;used$
850 ENDPROC
860 :
870 DEFPROCselect
880 choice=FNgetgridno(x%,y%,basex,bas
ey,7*32+8,2*32,5,2)
890 CASE choice OF
900 WHEN 1:IF frametot>0 CASE TRUE OF
910 WHEN x%<140:PROCframeone:PROCmo
usewait(0)
920 WHEN x%<208:PROCfradv
930 OTHERWISE PROCreplay(FALSE):PROC
mousewait(0)
940 ENDCASE
950 CALL code+4
960 WHEN 2:IF frametot>0 CASE z% OF
970 WHEN 2:PROCcycle:CALL code+4
980 WHEN 4:PROCreplay(TRUE):CALL co
de+4
990 ENDCASE
1000 WHEN 3:PROCbrush
```



```

1010 WHEN 4:
1020 WHEN 5:IF x%<1136 PROCload ELSE
PROCstar
1030 WHEN 6:PROCclear
1040 WHEN 7:PROCspeed(TRUE)
1050 WHEN 8:
1060 WHEN 9:
1070 WHEN 10:PROCsaved
1080 ENDCASE
1090 IF choice<>1 THEN PROCmousewait(0)
1100 ENDPROC
1110 :
1120 DEFPROCspeed(adjust)
1130 IF adjust THEN
1140 CASE z% OF
1150 WHEN 4:speed%=-1+8*(speed%=0):SO
UND 1,-10,70,1
1160 WHEN 2:speed%+=1:speed%=speed% M
OD 8:SOUND 1,-10,70,1
1170 ENDCASE
1180 ENDLF
1190 PRINTTAB(14,29);
1200 IF speed%>5 PRINT" ";
1210 IF speed%=0 PRINT"++"; ELSE PRINT;
50 DIV speed%;
1220 ENDPROC
1230 :
1240 DEFPROCcycle
1250 PROCmousewait(0)
1260 REPEAT
1270 MOUSE x%,y%,z%
1280 PROCreplay(TRUE)
1290 UNTIL z%>0:SOUND 1,-10,70,1
1300 ENDPROC
1310 :
1320 DEFPROCclear
1330 COLOUR 3 TINT 0
1340 PRINTTAB(2,29);"Confirm":VDU7
1350 COLOUR 0 TINT 0
1360 PROCmousewait(0):PROCmousewait(-1)
1370 IF z%=4 THEN
1380 startcol=col:starttint=tint
1390 PROCframeone
1400 frametot=0
1410 ENDLF
1420 PRINTTAB(2,29);"Clear "
1430 ENDPROC
1440 :
1450 DEFPROCdraw
1460 CASE brush OF
1470 WHEN 0:POINT x%,y%
1480 WHEN 1,2,3,4:CIRCLE FILL 4*(x% DI
V 4),4*(y% DIV 4),(brush+1)^2
1490 WHEN 5,6,7:n=3*(brush-3)^2:FOR Z%
=1 TO n/3:POINT x%-n/3+RND(n),y%-n/3+RND
(n)
1500 ENDCASE
1510 ENDPROC
1520 :
1530 DEFPROCbrush
1540 CASE z% OF
1550 WHEN 4:brush+=1:brush=brush MOD 8
:SOUND 1,-10,70,1
1560 WHEN 2:brush-=1+8*(brush=0):SOUND
1,-10,70,1
1570 ENDCASE
1580 COLOUR 0 TINT 0
1590 PRINTTAB(22,27);brush
1600 ENDPROC
1610 :
1620 DEFPROCframeone
1630 GCOL 128+startcol TINT starttint
1640 CLG:CALL code+4
1650 frameno=1:A%=screens+12:B%=A%
1660 ENDPROC
1670 :
1680 DEFPROCfradv
1690 *FX21,9
1700 IF frameno>frametot THEN
1710 VDU 7:PROCmousewait(0)
1720 ELSE
1730 A%=B%:frameno+=1
1740 WAIT:B%=USR(code+12)
1750 *FX15
1760 zz=INKEY(10)
1770 IF B%>0 THEN
1780 A%=B%
1790 ELSE VDU7
1800 ENDLF
1810 ENDLF
1820 ENDPROC
1830 :
1840 DEFPROCtopmouse
1850 CASE z% OF
1860 WHEN 4:GCOL col TINT tint:PROCdraw
1870 WHEN 2:col=POINT(x%,y%):tint=TINT
(x%,y%):PROCshowbox(col,tint)
1880 WHEN 1:PROCmakeframe:PROCmousewai
t(0)

```

Continued on page 27

New



**ON
RELEASE
SOON!**

ARCHWAY

Have you
spent long hours struggling
with the programming complexities of the brilliant
Archimedes **WIMPS** system?

With **ARCHWAY** that's all behind you. Now **YOU** can easily and quickly build multi-window programs with pop-up menus, icons, mouse control, etc. of a professional quality.

ARCHWAY provides tools, bricks, mortar and a basic structure. **YOU** slot the final bricks into place and add finishing touches.

A comprehensive suite of integrated tools lets you define and edit windows, menus, icons, dialogue boxes, mouse pointers, sprites, fill and line patterns, anti-aliased fonts, short cut keys and much more! You can import files (eg ARM machine code and music editor).

The bricks include an extensive library of functions and procedures providing ready access to many of Archimedes' most powerful features.

An in-depth user guide (300+ pages, ring bound) takes you gently step-by-step through a progressive series of program building examples with the emphasis on clarity.

The **ARCHWAY** run-time package together with script shells in BBC BASIC provide the structure.

£79.95
incl. VAT & p/p

The complete system comes on 4*800k discs (3 tools & 1 run-time). You need 1M of RAM to run the tools. One disc drive is sufficient.

SPECIAL INTRODUCTORY OFFER: No charge for upgrade to the extended RISC OS (Arthur 2) version we will release in 1989.



SIMTRON

*Programs to
help you*

**4 CLARENCE DRIVE, EAST GRINSTEAD,
WEST SUSSEX RH19 4RZ Telephone (0342) 328188**

Cheques/POs/official orders or Access/Visa number and expiry date.
24-hour 'phone for Credit Card orders.

ARE YOU GOOD ENOUGH?

As the leaders in software for the Archimedes range of computers, CLARES MICRO SUPPLIES are looking to extend our range even further. We are looking for people who are as excited by the Archimedes as we are.

If you have written any programs, completed or not, then we would like to hear from you.

If you have any ideas for programs and have the ability to execute the ideas then we want to hear from you.

If you have the ability to program the Archimedes but not the ideas to program then we want to hear from you.

Programs can be written in any language as long as they perform their stated task. Many of our programs contain large chunks of BASIC with ARM code in the areas that it is needed. BASIC on the Archimedes is a very powerful language and we do not attach any snob value to its use. If your program does what is meant to do then that's all we are interested in. Why not join the top team on the Archimedes. You get the support of our in-house team, privileged access through us to Acorn and invitations to our informal programmers seminars.

The most important point is that you will be earning top royalty rates if you prefer we will purchase your program outright.

Please write, in confidence, to Mr. D. Clare at:

**Clares Micro Supplies,
98 Middlewich Road,
Northwich,
CHESHIRE CW9 7DA**

If you have a program either complete or in development then please enclose a copy for our evaluation.

To protect yourself we advise that you lodge a copy of the program with your bank or solicitor BEFORE you send us a copy. You can then prove that your program pre-dates anything that we have.

Act today and become part of the leading software team producing software for the world's fastest micro.



MOVIE MAKER (continued from page 24)

```
1890 ENDCASE
1900 ENDPROC
1910 :
1920 DEFPROCcolours
1930 VDU26
1940 CASE z% OF
1950 WHEN 4:col=POINT(x%,y%):tint=TINT
(x%,y%):PROCshowbox(col,tint)
1960 ENDCASE
1970 VDU 24,basex+4;300+8;basex+1170;30
0+634;
1980 ENDPROC
1990 :
2000 DEFPROCshowbox(colour,tint)
2010 VDU26:GCOL colour TINT tint
2020 RECTANGLE FILL 1128,cpy+8,64,64
2030 VDU 24,basex+4;300+8;basex+1170;30
0+634;
2040 ENDPROC
2050 :
2060 DEFPROCmakeframe
2070 SOUND 1,-10,70,1:nextaddr=A%
2080 B%=USR(code+8)
2090 IF B%=0 THEN
2100 ERROR 255,"Delta File cache full"
2110 ELSE
2120 A%=B%:REM A% holds latest addr
2130 CALL code+4:REM copy to shadow
2140 frametot=frameno:frameno+=1
2150 ENDIF:ENDPROC
2160 :
2170 DEFPROCreplay(slow)
2180 GCOL 128+startcol TINT starttint
2190 CLG:B%=screens+12:N%=0:*FX200,1
2200 REPEAT
2210 A%=B%:N%+=1:B%=USR(code+12)
2220 IF slow AND speed%>0 THEN
2230 FOR S%=1 TO speed%
2240 WAIT
2250 NEXT
2260 ENDIF
2270 UNTIL N%=frametot OR B%=0
2280 IF B%>0 THEN A%=B%:frameno=N%+1 EL
SE frameno=N%
2290 *FX200
2300 ENDPROC
2310 :
2320 DEFPROCsaved
2330 PROCdibox("Save File")
2340 PROCdialogue:*FX200,1

2350 !screens=frametot
2360 screens!4=0:screens!8=0
2370 screens?4=startcol
2380 screens?5=starttint
2390 OSCLI("SAVE "+file$+" "+STR$~scree
ns+" "+STR$~A%)
2400 *FX200
2410 VDU26:PROCscreenssetup(FALSE)
2420 ENDPROC
2430 :
2440 DEFPROCload
2450 PROCdibox("Load File")
2460 PROCdialogue:*FX200,1
2470 OSCLI("LOAD "+file$+" "+STR$~scree
ns)
2480 frametot=!screens
2490 startcol=screens?4
2500 starttint=screens?5
2510 VDU26:PROCscreenssetup(FALSE)
2520 PROCframeone:*FX200
2530 ENDPROC
2540 :
2550 DEFPROCstar
2560 REPEAT
2570 PROCdibox("Star Command")
2580 INPUT"*"star$
2590 IF star$<>"" THEN
2600 CLS:VDU14
2610 OSCLI(star$)
2620 PRINT"Press any key or click";
2630 PROCmousewait(-2)
2640 ENDIF
2650 UNTIL star$=""
2660 MOUSE ON:OFF
2670 VDU26,15:PROCscreenssetup(FALSE)
2680 ENDPROC
2690 :
2700 DEFPROCdibox(text$)
2710 *FX15
2720 VDU26:MOUSE OFF:ON
2730 VDU 28,2,29,37,24
2740 COLOUR 128+57 TINT 192:COLOUR 0 TI
NT 0
2750 PROCplinth(basex-12,basex-12,35*32
+24+40,128+24+100,12,63+192,62+192,57+19
2,52,0,TRUE)
2760 PRINTtext$
2770 ENDPROC
2780 :
2790 DEFPROCdialogue
```



```

2800 PRINT "Current name ";file$
2810 INPUT "Filename : "input$
2820 IF input$<>" " THEN file$=input$
2830 REM COLOUR 128+57 TINT 192:COLOUR
0 TINT 0
2840 MOUSE ON:OFF
2850 ENDPROC
2860 :
2870 DEFPROCerror
2880 VDU7:*FX200,0
2890 PROCdibox("ERROR:")
2900 REPORT:PRINT" at line ";ERL
2910 PRINT!"To Quit Press Q"
2920 PRINT"To Continue, Press select";
2930 PROCmousewait(0):PROCmousewait(-2)
2940 IF w%=81 OR w%=113 THEN MODEL2:END
2950 VDU26:PROCscreensetup(FALSE)
2960 MOUSE ON:OFF
2970 ENDPROC
2980 :
2990 DEFPROCplinth(X,Y,WX,WY,W,C0,C1,C2
,C3,C4,toplight)
3000 IF toplight=FALSE THEN SWAP C1,C3
3010 GCOL C1 MOD 64 TINT C1 AND 240:REC
TANGLE FILL X,Y,WX,WY
3020 GCOL C3 MOD 64 TINT C3 AND 240:REC
TANGLE FILL X+W,Y,WX-2*W,W
3030 RECTANGLE FILL X+WX-W,Y,W,WY-W
3040 MOVE X,Y:MOVE X+W,Y:PLOT85,X+W,Y+W
3050 MOVE X+WX-W,Y+WY-W:MOVE X+WX,Y+WY-
W
3060 PLOT85,X+WX,Y+WY
3070 PLOT85,X+WX,Y+WY
3080 IF NOT toplight SWAP C0,C4
3090 GCOL C0 MOD 64 TINT C0 AND 240:LIN
E X,Y+WY,X+W,Y+WY-W
3100 GCOL C4 MOD 64 TINT C4 AND 240:LIN
E X+WX,Y,X+WX-W,Y+W
3110 GCOL C2 MOD 64 TINT C2 AND 240
3120 RECTANGLE FILL X+W,Y+W,WX-2*W,WY-2
*W
3130 ENDPROC
3140 :
3150 DEFPROCmousewait(n)
3160 w%=06
3170 REPEAT
3180 MOUSE x%,y%,z%
3190 IF n=-2 THEN w%=INKEY(0)
3200 UNTIL z%=n OR (n<0 AND z%>0) OR (n
=-2 AND w%>-1)
3210 ENDPROC
3220 :
3230 DEFPROCdrawgrid(basex,basey,cellx,
celly,nx,ny,xoff,yoff,boxcol,txcol)
3240 LOCAL lenx,leny,x,y
3250 lenx=cellx*nx:leny=celly*ny
3260 GCOL boxcol MOD 64 TINT boxcol AND
192
3270 FOR y=basey TO basey+leny STEP cel
ly
3280 LINE basex,y,basex+lenx,y
3290 NEXT
3300 FOR x=basex TO basex+lenx STEP cel
lx
3310 LINE x,basey,x,basey+leny
3320 NEXT
3330 RESTORE:VDU5
3340 GCOL txcol MOD 64 TINT txcol AND 1
92
3350 FOR y=basey+leny TO basey+celly ST
EP -celly
3360 FOR x=basex TO basex+lenx-cellex ST
EP cellx
3370 READ text$:MOVE x+xoff,y-4-yoff
3380 PRINTtext$
3390 NEXT:NEXT:VDU4:OFF
3400 ENDPROC
3410 :
3420 DEFFNgetgridno(x%,y%,basex,basey,c
ellx,celly,nx,ny)
3430 LOCAL lenx,leny,X,Y,no
3440 lenx=cellx*nx:leny=celly*ny
3450 IF FNpositiond(x%,y%,basex,basey,l
enx,leny) THEN
3460 X=(x%-basex)/DIVcellx+1
3470 Y=(y%-basey)/DIVcelly-1
3480 no=X+nx*Y
3490 ELSE no=-1
3500 ENDIF
3510 =no
3520 :
3530 DEFFNpositiond(x%,y%,xl,y1,x2,y2)
3540 =x%>xl AND x%<xl+x2 AND y%>y1 AND
y%<y1+y2
3550 :
3560 DEFFNpositiona(x%,y%,xl,y1,x2,y2)
3570 =x%>xl AND x%<x2 AND y%>y1 AND
y%<y2

```

PUBLIC DOMAIN SOFTWARE FOR THE ARC

There is plenty of public domain software around for PC compatibles, but as Lee Calcraft reports, there is now a trickle for the Arc.

The idea of public domain software is an excellent one. It relies on benevolent authors foregoing the copyright on their creations, and permitting them to be sold openly for a nominal fee. If you scan the pages of PCW, or better still, American journals such as Byte, you will see hordes of public domain software for PC compatibles advertised.

It is good to report that there is now a small number of titles available for the Arc. The common factor in all this is the C programming language. All of the Arc public domain software which I have seen has been originally written for other computers in C, and has been compiled into ARM code, and appropriately modified to run on the Arc.

The two inserts give a list of titles currently available with price and source. First of all, let me mention the *Arc Archiver*. This was distributed on the magazine disc for RISC User Issue 8. It allows you to archive large numbers of files in a compressed form. It is suitable for storing program and text backups in the minimum of space, and really comes into its own if you are transferring files via a modem. Programs and text files can typically be compressed to 50-60% of their normal size. As with all public domain software, the instructions are supplied on the disc itself.

Aim is also well worth a mention. This contains a massive suite of image processing software, which works through the WIMP system. It is really very impressive to use, though I should point out that it will only work in mode 20, so you will need access to a multi-sync monitor.

Arc	-	File archiver and compressor (1)
Aim	-	Image processing software (2)
Micro Spell	-	43,000 word spell check
Kermit	-	Communications software
C Toolkit	-	Utilities for C
EMACS	-	Multiple file text editor
WIMP Chess	-	WIMP-based chess
XLISP	-	Object orientated LISP language
Cross Star	-	WIMP-based crossword solver with 100,000 word dictionary

Finally, you will see that the majority of titles here are supplied by David Pilling (who in fact wrote Beebug's *Hearsay* suite, and who also converted the *Arc Archiver*). Worthy of mention in this group is a WIMP-based chess program, which, as far as I can see from brief tests, seems to play a very intelligent game indeed. There is also a spelling checker with a 43,000 word dictionary, a C toolkit, a complete Kermit comms terminal, and a very sophisticated text editor.

Every one of these titles has to be an absolute bargain at the price listed, and we will be covering some of them in greater detail in future issues. In the mean time, if you know of other public domain software for the Arc, please let us know; and if you have a C compiler, why not try making some available yourself?

RU

Suppliers:

(1) *RISC User magazine disc, Volume 1, Issue 8.*
(2) *Lingenault (E6), P.O.Box 10, Halesworth, Suffolk IP19 0DX*
The remainder (at £5.99 inc.VAT) from:
David Pilling, P.O. Box 22, Thornton Cleveleys, Blackpool, FY5 1LR.

Points Arising....Points Arising....Points Arising....Points Arising....

Supercharged RISC User Disc Menu (Vol.1 Issue 10 & Vol.2 Issue 1)

As it stands the RISC User disc menu will not work on a hard disc system if the disc has no name.

This is easily remedied by changing the two lines below and reassembling the module.

1480.1bl1 SUBS R3,R3,#1:MOVMI PC,R14

1490 LDRB R0,[R2,#1]!:SWI wc:B 1bl1

This also applies to the Disc Menu included on the current version of the *RISC User Volume One Special Disc*. Version 2 of the disc, however, will have these changes incorporated.

RU

Reviewed by Mike Williams.

When I reviewed Pipedream for the Archimedes (RISC User Volume 1 Issue 8) I said then that a major omission was that of a spelling checker. Colton Software readily acknowledge the demand for a facility of this kind, and made it known at an early stage that such an extension was considered to be of the highest priority.

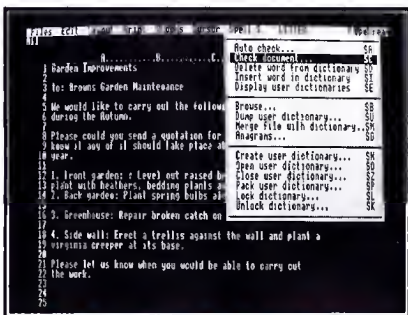
Well, the spelling checker (called simply Pipedream Spellcheck) is here, and it includes an 80,000 word dictionary occupying just over 0.25Mbytes. The package consists of a single disc plus a 48 page user guide with Colton Software's distinctive dark blue cover.

The first task is to copy the dictionary and other files onto your existing working copy of Pipedream, which must then be re-installed to enable the Spellcheck option. Despite several pages of help and advice, I still found the process more confusing than I would wish. And even when properly installed, your Pipedream disc will not automatically auto-boot unless you choose (and know how) to set it up in this way.

Once Pipedream has been loaded, the same familiar screen appears, but with an additional *Spell* option in the menu bar. Pressing Alt-S produces a pull-down menu with all the Spellcheck options. Like many spelling checkers now available, Pipedream's version may be used in immediate mode to check the spelling of words as they are typed in, or subsequently to check all or part of an existing document.

In immediate mode, each word is checked as the terminating space or punctuation symbol is entered. If the word cannot be matched in the dictionary a bleep sounds. By default the dictionary is accessed from disc, and despite obvious optimisation there is a definite tendency for the spelling checker to lag behind even my modest two-finger typing. However, I must be honest and confess I have never found much use personally for this facility, preferring to check a complete document once all the writing and editing has been completed. But if you need it, then it's there.

Pipedream clearly loads sections of the dictionary into free memory to minimise disc accessing, and you can even lock the entire dictionary into memory, if enough free space is available. I found on a 310 that promises well over half a megabyte on power-up, that there was still insufficient room for the whole dictionary even with only a few lines of text typed in. Because of its optimised approach to the use of free memory, the locked dictionary option may be both as unnecessary as it appears impracticable.



The Pipedream Spellcheck menu

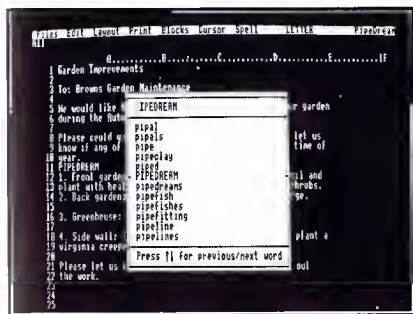
A *Browse* option allows you to peruse the dictionary at any time, starting at the current word. Wild card characters can also be incorporated to aid searches for words to fit a specific pattern. The manual warns here, that very restrictive patterns may result in some time elapsing before any words can be matched in the dictionary, if any. If *Browse* mode has been selected, then any word accepted from the dictionary is inserted into the text at the current cursor position, but without replacing the current word.

However, if the *Check document* option has been selected things work differently. Any word not matched is shown in a dialogue box with essentially four choices available to the user. A word can be ignored (it is acceptable even though not in a dictionary). Unlike some spelling

checkers, once a word has been accepted, subsequent occurrences will no longer be flagged while the same document is active. You can change the word to whatever you choose, or you can browse through the dictionary for the correct spelling (and any word selected will now *replace* the current text word), and finally the word may be added to a user dictionary.

USER DICTIONARIES

The majority of the options in the Spellcheck pull-down menu are concerned with user dictionaries. You can create as many of these as you wish, and up to five may be open at any time, though most users will probably just use one dictionary of their own. Pipedream automatically includes any user dictionaries which are open when checking spellings.



Browsing through the Spellcheck dictionary

The first step is to create a user dictionary in an appropriate directory, and then open this dictionary, both options in the Spellcheck menu. You can also check which user dictionaries, if any, are open. Words may be explicitly added to or deleted from user dictionaries, as well as adding words (when appropriate) which are not matched in the main dictionary or any open user dictionary.

There are some limits on what Pipedream will accept as a 'word', no digits are allowed for example, but hyphens are. In fact, when replacing words not matched by a dictionary, the offending word can be replaced by one

which is hyphenated, or by one which now includes a space (thereby becoming two words). Again, I find this to be a good practical approach, but one missing in some other spelling checkers. However, Pipedream does not check the spelling of replacement words input by the user.

Dictionaries, particularly user dictionaries, do require some care in their use. Clearly Pipedream must be able to access any dictionaries when required, even if you keep your text files on other discs (a good reason for locking the dictionary into memory if you've got the space). If you do have the wrong disc inserted the result is just mildly inconvenient rather than catastrophic. Browsing through the dictionary supplied reveals the usual amazing collection of unheard of words, and Pipedream is included even if RISC isn't.

CONCLUSIONS

I quite enjoy Pipedream for its various capabilities, and the addition of a spelling checker will enhance its appeal as a word processor. The new facility integrates well with Pipedream, and is generally as workmanlike and easy to use as might be expected. Whether a disc-based dictionary will be fast enough for your needs only practice will really tell - I found it to be no great limitation. Given the memory of a 440, a locked dictionary is the way to go.

Overall, there is very little to complain about, and plenty to be grateful for. There is little more to say.

**Product
Supplier**

**Pipedream Spellcheck
Colton Software
Broadway House,
149-151 St Neots Road,
Hardwick,
Cambridge CB3 7QJ.
Tel. (0954) 211472
£49.45 Inc VAT**

Price

Note: the Spellcheck option will only work with version 2.2 and later of Pipedream. Users of earlier versions can obtain a free upgrade from Colton software by returning their original Pipedream disc.

RU

PipeDream

PipeDream is now available on the Acorn Archimedes. It provides comprehensive word processing, spreadsheet and database facilities integrated in a way only dreamed of before.

With other integrated packages, you have to divide your work into artificial sections, such as text, numbers and calculation, and database.

With PipeDream, you compose your work in the order you want to print it, with text and numbers all together in one document. Incorporate calculations directly into paragraphs of text and formatted paragraphs directly into spreadsheets.

PipeDream is ideal for all tasks involving words and numbers. From writing thank-you letters to encyclopaediae, invoices to cash flow forecasts, stamp-collection records to inventory management, film scripts to mail-shots.

PipeDream is 100% file and keystroke compatible with Z88 PipeDream and PipeDream on the IBM PC. It is also compatible with View Professional on the BBC microcomputer. You can create documents on any of these computers, transfer the files to any other and continue working, with no loss of information. No other software enables you to share your files with all these computers.

PipeDream for the Acorn Archimedes costs £99 + VAT.

It is not possible to detail all of PipeDream's facilities here. For full details or to order PipeDream call us on 0954 211472 or write to us at Colton Software, Broadway House, 149-151 St Neots Road, Cambridge CB3 7QJ.

PipeDream - power at your fingertips.



RISC USER TOOLBOX (6)

David Spencer adds to the RISC User Toolbox a pot-pouri of commands suggested by members.

There are four star commands added to the Toolbox this month. Two of these are for printing out memory dumps and disassemblies, one is for comparing two files, and the final one is for compacting ADFS discs.

ENTERING THE LISTING

As with previous months, the lines given in the listing below should be added to the complete Toolbox source code. This should be relatively easy, because apart from the command table and help messages, this month's new lines are in a single block. Incidentally, if you are using the source code program from the magazine disc you will notice that it contains a few extra lines. These merely select mode 0 at the start, and prompt before saving the Toolbox module. These can be ignored or deleted as required.

Once the new listing has been entered, it should be saved under a different name from the original source code, and run. This will cause the complete Toolbox module to be assembled and saved to disc. To load the Toolbox module type:

```
QUIT
TOOLBOX
```

The four new commands, which can be found using:

```
*HELP COMMANDS
are *MDUMP, *DDUMP, *COMPARE and
*DCOMPACT.
```

*MDUMP AND *DDUMP

These two commands are used to print out either a dump of an area of memory, using *MDUMP or a disassembled listing of an area with *DDUMP.

Both commands take the same parameters:

```
*MDUMP <start> <end> [<offset>]
*DDUMP <start> <end> [<offset>]
```

Where, <start> is the start address of the area in hex, and <end> is the end address. As an alternative to giving the end address, the length of the area can be given by using a '+' sign. For example:

```
*MDUMP 9000 +1000
```

Both commands produce their output in the same format as *MEDIT and *DISASS described previously, except that no colours are used. This prevents any problems when outputting to a printer.

The optional <offset> parameter allows the address printed in the output to be different from the first address. For example:

```
*MDUMP 3800000 +1000 10000
```

will dump out the 4096 byte block of memory starting at &3800000, but the addresses printed at the start of each line will start at &10000. In the case of *DDUMP, any branches and program counter relative instructions will be displayed relative to the offset address rather than the actual address. This is most useful when printing out the disassembled listing of a non-relocatable ARM program that cannot be loaded at its genuine execution address. For example, a compiled C program loads into memory starting at location &8000, and cannot therefore be loaded while Basic is active (because Basic uses location &8000 upwards as workspace).

Both the start and offset addresses are rounded down before the dump is produced. In the case of *DDUMP, they are rounded down to a word boundary, while for *MDUMP they are rounded down to the start of a sixteen byte block. Similarly, the end address is rounded up if it is not already on a word (*DDUMP) or sixteen byte (*MDUMP) boundary.

To send the output of these commands to a printer simply press Ctrl-B before pressing Return at the end of the command line, and Ctrl-C once the command has finished printing. Incidentally, to stop a prompt symbol appearing at the end of the printed output, press Ctrl-A and Delete before the Ctrl-C.

*COMPARE

This command performs a simple comparison of two files. The syntax is:

```
*COMPARE <1st file> <2nd file> [<max diff>]
```

<1st file> and <2nd file> are the names of the two files to be compared.



The comparison is on a byte by byte basis, with any differences being reported in the form:

```
000000: aa b cc d
```

Where, 000000 is the position within the file in hex, aa is the byte from the first file in hex, b is the equivalent ASCII character, cc is the byte from the second file, and d is its corresponding ASCII code. If the two files are of different lengths, this is be reported, and the comparison will stop at the end of the shorter of the two files. An error is generated if either file can't be found.

The optional <max diff> parameter allows you to specify the number of discrepancies between the files after which to terminate the command. By default, up to sixteen differences are allowed before the command is aborted. However, this can be over-ridden by giving an alternative value (in decimal). To see all the differences between the files use a value of 0, for example:

```
*COMPARE File1 File2 0
```

*DCOMPACT

The final new command performs a similar function to the ADFS command *COMPACT, except that it ensures that all the free space on the disc is collected into a single block. *COMPACT merely ensures that the free space fragmentation is reduced. The syntax for *DCOMPACT is:

```
*DCOMPACT <disc spec.>
```

In contrast to *COMPACT, the disc spec. must be given, for example:

```
*DCOMPACT :0
```

or *DCOMPACT :TechDoc

When *DCOMPACT is executing, it prints the message 'Compacting' followed by a series of dots until the process is finished. The command works by repeatedly calling *COMPACT until the size of the largest free block on the disc is the same as the total free space. This means that only one block of free space then exists.

```
335 EQU "Mdump":EQUB 0
336 ALIGN:EQUd mdumpc:EQUd &30002
337 EQUd mdsyn:EQUd mdhlp
338 EQU "Ddump":EQUB 0
339 ALIGN:EQUd ddumpc:EQUd &30002
340 EQUd ddsyn:EQUd ddhlp
```

```
341 EQU "Compare":EQUB 0
342 ALIGN:EQUd compc:EQUd &30302
343 EQUd cosyn:EQUd cohlp
344 EQU "DCompact":EQUB 0
345 ALIGN:EQUd dcomc:EQUd &10001
346 EQUd dcomsyn:EQUd dcomhlp
1405 .mdhlp EQU "Mdump dumps the contents of an area of memory in hex and ASCII.":EQUB 13
1406 .mdsyn EQU "Syntax: Mdump <start add> [+<end add>|length> [<offset>]":EQUB 0:ALIGN
1407 .ddhlp EQU "Ddump dumps the disassembled contents of an area of memory.":EQUB 13
1408 .ddsyn EQU "Syntax: Ddump <start add> [+<end add>|length> [<offset>]":EQUB 0:ALIGN
1409 .cohlp EQU "Compare performs an unintelligent comparison between two files.":EQUB 13
1410 .cosyn EQU "Syntax: Compare <file 1> <file2> [<max.diff>]":EQUB 0:ALIGN
1411 .dcomhlp EQU "Dcompact compacts an ADFS disc until all the free space is in a single block.":EQUB 13
1412 .dcomsyn EQU "Syntax: Dcompact <disc spec.>":EQUB 0:ALIGN
11890 .ddumpc
11900 STMFD R13!,{R14}:LDR R12,[R12]
11910 BL getarg
11920 BIC R0,R0,#3:ADD R1,R1,#3:BIC R1,R1,#3
11930 BIC R2,R2,#3:SUB R2,R2,R0:MOV R3,R1
11940 MOV R1,#9
11950 .ddumpc2 STMFD R13!,{R0-R2}
11960 BL swi9:BL prtnoc
11970 SWI "OS NewLine"
11980 LDMFD R13!,{R0-R2}
11990 SWI "OS_ReadEscapeState":LDMCSFD R13!,{PC}
12000 ADD R0,R0,#4:CMR R0,R3
12010 BNE ddumpc2:LDMFD R13!,{PC}
12020 :
12030 .mdumpc
12040 STMFD R13!,{R14}:LDR R12,[R12]
12050 BL getarg:BIC R0,R0,#15
12060 ADD R1,R1,#15:BIC R1,R1,#15
12070 BIC R2,R2,#15:SUB R2,R2,R0
12080 MOV R3,R1:MOV R1,#1
12090 STR R1,[R12,#32]:STR R2,[R12,#12]
12100 .ddumpc2 STMFD R13!,{R0}
12110 MOV R0,#&1B:ADR R1,wrin
```



RISC USER TOOLBOX

```
12120 ADD R2,R12,#256:SWI "OS_Claim"
12130 MOV R0,#0:STR R0,[R2]
12140 MOV R0,#3:MOV R1,#&74:SWI "OS_Byte"
"
12150 LDMFD R13!,{R0}:STMF R13!,{R1}
12160 BL prtlne
12170 LDMFD R13!,{R1}:STMF R13!,{R0}
12180 MOV R0,#3:SWI "OS_Byte"
12190 MOV R0,#&1B:ADR R1,wrin
12200 ADD R2,R12,#256:SWI "OS_Release"
12210 ADD R2,R2,#1
12220 BL prtnoc:LDMFD R13!,{R0}
12230 SWI "OS_NewLine"
12240 ADD R0,R0,#16:CMR R0,R3
12250 LDMEQFD R13!,{PC}
12260 SWI "OS_ReadEscapeState"
12270 BCC mdumpc2:LDMFD R13!,{PC}
12280 .wrin
12290 STMF R13!,{R1}
12300 LDRB R1,[R12]:ADD R1,R1,#1
12310 STRB R0,[R2,R1]
12320 STRB R1,[R12]
12330 LDMFD R13!,{R1}:MOV PC,R14
12340 :
12350 .prtnoc
12360 STMF R13!,{R14}
12370 .prtnoc2 LDRB R0,[R2],#1
12380 CMP R0,#0:CMPE R0,#10:LDMEQFD R13!,{PC}
12390 CMP R0,#17:ADDEQ R2,R2,#1
12400 BEQ prtnoc2:SWI "OS_WriteC"
12410 B prtnoc2
12420 :
12430 .getarg
12440 STMF R13!,{R3-R5,R14}:MOV R5,R1
12450 MOV R1,R0:MOV R0,#16:SWI "OS_ReadUnsigned"
12460 SUB R13,R13,#12:STR R2,[R13]:MOV R3,R2
12470 .getarg2 LDRB R4,[R1,#1]:CMP R4,#32
12480 BEQ getarg2:CMR R4,#ASC+":ADDEQ R1,R1,#1
12490 MOV R0,#16:SWI "OS_ReadUnsigned"
12500 CMP R4,#ASC+":ADDEQ R2,R2,R3
12510 STR R2,[R13,#4]:CMR R5,#2
12520 MOVEQ R2,R3:BEQ getarg4
12530 .getarg3 LDRB R4,[R1,#1]:CMP R4,#32
12540 BEQ getarg3:MOV R0,#16
12550 SWI "OS_ReadUnsigned"
12560 .getarg4 STR R2,[R13,#8]
12570 LDMFD R13!,{R0-R5,PC}
12580 :
12590 .compc
12600 STMF R13!,{R14}:LDR R12,[R12]
12610 MOV R3,R1:MOV R1,R0
12620 MOV R0,#&40:MOV R5,R1
12630 SWI "OS_Find":CMR R0,#0
12640 ADREQ R0,srcerr:LDMEQFD R13!,{R14}
12650 ORREQS PC,R14,#1<<28
12660 MOV R4,R0:MOV R0,R5
12670 MOV R2,#1<<29:SWI "OS_GSInit"
12680 .compc2 SWI "OS_GSRead":BCC compc2
12690 MOV R5,R0:MOV R1,R0
12700 MOV R0,#&40:SWI "OS_Find"
12710 CMP R0,#0:ADREQ R0,dsterr
12720 LDMEQFD R13!,{R14}
12730 ORREQS PC,R14,#1<<28
12740 MOV R1,R5:MOV R5,R0:MOV R0,R1
12750 CMP R3,#2:MOVEQ R3,#16:BEQ compc5
12760 MOV R0,#1<<29:SWI "OS_GSInit"
12770 .compc3 SWI "OS_GSRead":BCC compc3
12780 .compc4 LDRB R1,[R0]:CMP R1,#32
12790 ADDEQ R0,R0,#1:BEQ compc4:MOV R1,R0
12800 MOV R0,#10:SWI "XOS_ReadUnsigned"
12810 MOV R3,R2:BVC compc5:MOV R3,R0
12820 MOV R0,#0:MOV R1,R4:SWI "OS_Find"
12830 MOV R0,#0:MOV R1,R5:SWI "OS_Find"
12840 MOV R0,R3:LDMEQFD R13!,{R14}
12850 ORRS PC,R14,#1<<28
12860 .compc5 MOV R0,#2:MOV R1,R4
12870 SWI "OS_Args":MOV R6,R2
12880 MOV R0,#2:MOV R1,R5:SWI "OS_Args"
12890 CMR R2,R6:BEQ compc6:MOVCC R6,R2
12900 SWI "OS_Writes"
12910 EQUUS "Files are not the same length"
12920 EQUUB 10:EQUUB 13:EQUUB 0
12930 ALIGN
12940 .compc6 MOV R7,#0
12950 .compclloop MOV R1,R4:SWI "OS_BGet"
12960 MOV R8,R0:MOV R1,R5:SWI "OS_BGet"
12970 CMP R0,R8:BEQ compcmatch
12980 STMF R13!,{R0}:MOV R0,R7
12990 ADD R1,R12,#256:MOV R2,#10
13000 SWI "OS_ConvertHex6"
13010 SWI "OS_Write0":SWI (&100+ASC+":")
13020 SWI &120:SWI &120
13030 MOV R0,R8:ADD R1,R12,#256
13040 MOV R2,#10:SWI "OS_ConvertHex2"
13050 SWI "OS_Write0":MOV R0,R8
13060 BL prtchar:SWI &120
13070 SWI &120:LDMFD R13!,{R0}
13080 ADD R1,R12,#256:MOV R2,#10
13090 SWI "OS_ConvertHex2"
13100 SWI "OS_Write0":LDMFD R13!,{R0}
```


RISC USER TOOLBOX

MOUSE MENU

```

13110 BL prtchar:SWI "OS_NewLine"
13120 SUBS R3,R3,#1:BEQ compcout
13130 .compmatch ADD R7,R7,#1
13140 CMP R7,R6:BEQ compcout
13150 SWI "OS_ReadEscapeState"
13160 BCC compcloop
13170 .compcout
13180 MOV R0,#0:MOV R1,R4
13190 SWI "OS_Find"
13200 MOV R0,#0:MOV R1,R5
13210 SWI "OS_Find"
13220 LDMFD R13!,{PC}
13230 .srcerr
13240 EQU0 0
13250 EQU0 "First file not found"
13260 EQU0 0:ALIGN
13270 .dsterr
13280 EQU0 0
13290 EQU0 "Second file not found"
13300 EQU0 0:ALIGN
13310 :
13320 .prtchar STMFD R13!,{R14}
13330 CMP R0,#32:MOVLO R0,#ASC".
13340 CMP R0,#&7F:MOVHS R0,#ASC".
13350 SWI &120:SWI "OS_WriteC"
13360 LDMFD R13!,{PC}
13370 :
13380 .dcomc
13390 STMFD R13!,{R14}:LDR R12,[R12]
13410 ADD R2,R12,#264
13420 .dcomcop LDRB R3,[R0],#1
13430 STRB R3,[R2],#1:CMP R3,#13
13440 BNE dcomcop
13450 SWI "OS_Writes"
13460 EQU0 "Compacting ":EQU0 0
13470 ADR R0,comcom:ADD R2,R12,#256
13480 LDMIA R0,{R3,R4}
13490 STMIA R2,{R3,R4}
13500 .dcomcl SWI (&100+ASC".")
13510 ADD R0,R12,#256:SWI "OS_CLI"
13520 ADD R0,R12,#264
13530 SWI "ADFS_FreeSpace":CMP R0,R1
13540 BNE dcomcl:SWI "OS_NewLine"
13550 LDMFD R13!,{PC}
13560 .comcom
13570 EQU0 "COMPACT "

```

RU

MOUSE MENU (continued from page 14)

```

2690 :
2700 DEF FNwhich(ypos)
2710 LOCAL pos,line
2720 pos=(16-ypos) DIV 32
2730 line=-1
2740 REPEAT
2750 line+=1
2760 UNTIL start(line)>pos
2770 IF start(line)-pos=1 THEN =0 ELSE
=line
2780 :
2790 REM Sample Menu
2800 DATA .menu1
2810 DATA SAMPLE MENU
2820 DATA 1,First run
2830 DATA FIRST ENTRY
2840 DATA The first entry in the menu h
as both
2850 DATA "subsidiary text, and an info
rmation"
2860 DATA box.
2870 DATA !TEXT
2880 DATA You have just clicked on the
first
2890 DATA menu entry. This is the addit
ional
2900 DATA text for that entry.
2910 DATA !END
2920 DATA 1,Second Run
2930 DATA SECOND ENTRY
2940 DATA !TEXT
2950 DATA You have clicked on the secon
d entry
2960 DATA in the menu. This entry has n
o subsidiary
2970 DATA text and hence the menu entry
consists
2980 DATA of just the heading.
2990 DATA !END
3000 DATA -2,"SOUND 1 &FFFFFFF0 100 100
"
3010 DATA THIRD ENTRY
3020 DATA The third and final menu entr
y
3030 DATA "has some descriptive text, b
ut"
3040 DATA no information box. If you
3050 DATA double-click on this option a
3060 DATA "musical note will be played,
and"
3070 DATA an exit made from the menu.
3080 DATA !END
3090 DATA !MENUEND

```

RU

EXPANSION CARDS FOR THE ACORN ARCHIMEDES COMPUTER SYSTEM

IEEE488 INTERFACE a full implementation of the standard for automatic test and measurement systems

16 BIT PARALLEL I/O two 16 bit input or output ports with handshake lines for digital control applications

DUAL RS423 SERIAL INTERFACE for communicating with two additional RS423 or RS232 devices eg printers, plotters, instruments, etc

12 BIT ANALOGUE I/O in development

All the above high performance expansion cards are supplied with high level software for ease of use and a comprehensive user guide.

Take advantage of Intelligent Interfaces' expertise and purchase a complete Archimedes Computer System.

Officially appointed Acorn Scientific Dealer.



Intelligent Interfaces Ltd

43b Wood Street
Stratford-upon-Avon
Warwickshire
CV37 6JQ

Tel: 0789 415875

Telex: 312242 MIDTLX G



Archimedes Visuals

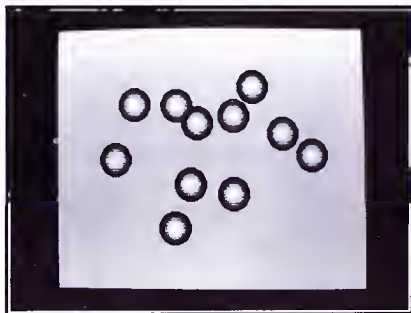
A clever animation using sprite masking techniques,
and a powerful 256 colour selector.

Both programs require 160K of screen RAM.
The first also requires 8K of sprite RAM

Masked Spheres

by Paul Jennings

This very clever Basic program creates a collection of eleven shaded 3D spheres, and rotates the cluster about the centre of the screen in real time. To achieve such impressive results Paul has used one or two tricks of the trade. Firstly he uses dual screens, so that the program can be drawing the next image while the observer is watching the one before it. This is achieved using FX112 and FX113. Secondly, the spheres are drawn as sprites. This gives an excellent speed advantage. Thirdly, as each is plotted, the order of plotting is chosen in such a way that the spheres "furthest" from the observer are drawn first. This creates the illusion of depth because the "nearer" spheres appear in front of the "further" ones.



Finally, he uses a sprite transparency mask. When you run the program, you will see the effect without the mask. Pressing the space bar creates the mask, and produces a perfect image. Without it, each sphere appears within a black filled square. The procedure PROCspritmask creates a transparency mask for the sprite. This is achieved using four sprite SYS calls. The first generates a blank mask for

the sprite called "sphere", while the second returns the sprite's width and height. This data is then used in a pair of FOR loops which scan the sprite. SYS46,41 reads the colour of the sprite at the specified point, and if this is zero, SYS46,44 is called to make the specified pixel transparent. The procedure should work without modification with any sprite, providing that you change references to the sprite name where appropriate.

```
10 REM >MskdSphrs
20 REM Program Masked Spheres
30 REM Version A 0.04
40 REM Author Paul Jennings
50 REM RISC User December 1988
60 REM Program Subject to Copyright
70 :
80 ON ERROR OSCLI("FX114,1"):MODE12:R
EPORT:PRINT;" at line ";ERL:END
90 *FX114,0
100 MODE13:OFF
110 DIMY(11),rad(11),S(360),C(360)
120 PROCsprites
130 PROCinit
140 GCOL8,0
150 COLOUR128+37 TINT192:CLS
160 :
170 REPEAT
180 FOR Ang=0 TO 359 STEP step
190 IF INKEY(-99) AND mask%=FALSE THEN
PROCspritemask
200 WAIT
210 OSCLI("FX113 "+D$)
220 OSCLI("FX112 "+U$)
230 CLS
240 IF Ang>180 THEN S%=1:E%=11:T%=1 EL
SE S%=11:E%=1:T%=-1
250 FOR L=S% TO E% STEP T%
260 IF L<7 THEN
270 x=rad(L)*S(Ang)
280 y=0.2*rad(L)*C(Ang)
290 ELSE
300 IF Ang<180 y=0.2*rad(L)*C(Ang+180)
:x=rad(L)*S(Ang+180)
310 IF Ang>=180 y=0.2*rad(L)*C(Ang-180)
):x=rad(L)*S(Ang-180)
```

```

320 ENDIF
330 ORIGIN 590,y(L)
340 PLOT&ED,x,y
350 NEXT
360 IF U$="0" U$="1" ELSE U$="0"
370 IF D$="0" D$="1" ELSE D$="0"
380 NEXT Ang
390 UNTIL FALSE
400 :
410 DEFPROCinit
420 mask%=FALSE:step=4:A=PI/2
430 D$="1":U$="0"
440 FOR L=0 TO 360
450 S(L)=SIN(A):C(L)=COS(A)
460 A=A+2*PI/360
470 NEXT
480 RESTORE
490 FOR L=1 TO 11
500 READ height,radius
510 y(L)=height:rad(L)=radius
520 NEXT
530 ENDPROC
540 :
550 DEFPROCsphere(col,X,Y)
560 T%=0
570 FOR radius=64 TO 8 STEP -8
580 GCOL col TINT T%
590 CIRCLE FILL X,Y,radius:T%=T%+64
600 IF radius=40 THEN col+=21:T%=0
610 NEXT
620 ENDPROC
630 :
640 DEFPROCsprites
650 PROCsphere(34,640,512)
660 ORIGIN0,0:MOVE576,448:MOVE704,576
670 *SGET sphere
680 ENDPROC
690 :
700 DEFPROCspitemask
710 mask%=TRUE
720 SYS46,29,0,"sphere",0,0
730 SYS46,40,0,"sphere",0,0 TO ,,,width,height
740 FOR x=0 TO width-1
750 FOR y=0 TO height-1
760 SYS46,41,0,"sphere",x,y TO ,,,,col,tint
770 IF col=0 AND tint=0 THEN SYS46,44,0,"sphere",x,y,0
780 NEXT:NEXT

```

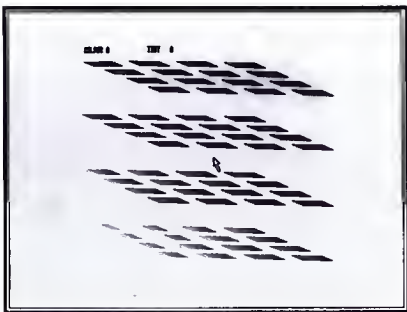
```

790 ENDPROC
800 :
810 DATA 700,400,500,500,200,200,670
820 DATA 190,590,85,360,120,600,100
830 DATA 300,100,500,350,400,500,700
840 DATA 200

```

3D 256 Colour Selector by T.K. Boyd

Selecting colours in the 256 colour modes is not an easy task. One of the problems is the layout of the colour chart itself. The only really logical way to do it is three-dimensionally, with the red, green and blue components increasing as you move along the three axes. T.K. Boyd's program achieves this, and makes it easy - or easier - to select sequences of related colours.



If you run the program, and move the pointer around, you will see that it displays the colour and tint number at the pointer. To select a linked sequence of colours (e.g. for different shades of a lighted object), imagine any straight line drawn through the 3D volume. The colours intersected will form a sequence of related hues.

```

10 REM >3DColSel
20 REM Program 256 Colour Selector
30 REM Version A 0.2
40 REM Author T.K. Boyd
50 REM RISC User December 1988
60 REM Program Subject to Copyright

```

```
70 :
80 MODEL5:OFF:*POINTER
90 PROC3d
100 PROCmouse
110 END
120 :
130 DEFPROCmouse
140 REPEAT
150 MOUSE x%,y%,z%
160 PRINTTAB(0,0)"COLOUR ";POINT(x%,y%
);" "
170 PRINTTAB(20,0)"TINT ";TINT(x%,y%
);" "
180 UNTIL z%=1
190 ENDPROC
200 :
210 DEFPROC3d
220 XDISP=0:YDISP=200
230 ScXX=232:ScYY=250:ScXZ=110:ScYZ=40
240 XREC=36:XREC2=55:YREC=20
250 FOR X=0 TO 3
260 FOR Y=0 TO 3
270 FOR Z=0 TO 3
280 MOVE XDISP+Z*ScXZ+X*ScXX,YDISP-Z*Sc
YZ+Y*ScYY
290 PROCDrawBlob(X,Y,Z)
300 NEXT:YREC:YREC2
310 ENDPROC
320 :
330 DEFPROCDrawBlob(R,G,B)
340 FOR Tint=0 TO 3
350 GCOL R+G*4+B*16 TINT Tint*64
360 PLOT1,XREC2,YREC*-1
370 PLOT1,XREC,0
380 PLOT1,XREC+XREC2)*-1,YREC
390 PLOT1,XREC,0
400 PLOT1,XREC2,YREC*-1
410 PLOT1,XREC2*-1,YREC
420 NEXT
430 ENDPROC
```

RU

Archimedes Shareware Discs

Graphics Demo 1 (£3.50)

49 graphics demo programs plus MenuMaster—a program to allow you to edit the demo order and add your own favourite demos.

Shareware Disc N° 1 (£3.50)

MenuMaster with seven more graphics demo programs plus Life, Mandelbrots, Structured Directory Lister, European Geography.

Shareware Disc N° 2 (£4.50)

DFS reader, backup and archiver. 9 Graphics demos. Connect four, Mastermind, Solitaire and StarTrek games. 256 colour Sprite Editor. CMOS ram Editor, Disc Editor, LQprinter font definer, Matrix Functions, Memory Mappings & Vector Listings BASIC fast Screenload.

**Save £1.50 – Buy all 3
for just £10.**

Also Available... WIMP Template Editor (£8)

Design your own windows, icons, menus and sprites – completely freehand in a wysiwyg format. Create WIMP templates which can easily be loaded from your own program – no more need for endless data blocks!!!

Based on the original "Form Editor" program from Acorn Computers, modified and extended (with Acorn's permission!)

Prices include UK p&p. European prices as UK. Outside Europe sent by surface but add £1 per order for airmail. Pay by UK cheque or Eurocheque or credit our Girobank account 2341107. Sorry no credit cards.

Norwich Computer Services, 18 Mile End Road, Norwich, NR4 7QY. (0603-507057)



Word Up Word Down

An innovative 3-dimensional word game.

- * 3-dimensional and plan view displays
 - * 33,000 + word dictionary
- * Professionally digitised speech and sound fx
 - * 1 to 6 players
- * Computer play options over 3 skill levels
 - * Rotating 3-dimensional board
- * Time limit plus other extensive options
- * User-definable palette and mouse options

Also Available... **DESKTOP GAMES (£5.95)** - 3 puzzles and 2 utilities for the desktop. Great for children using the mouse.

CHRISTMAS OFFER
 Buy **WORD UP WORD DOWN** and
STARTRADER together and get
DESKTOP GAMES



STARTRADER

A strategic space adventure of epic proportions

- * 800 unique planets spanning 112 star systems
- * Professionally digitised sound effects and speech
 - * 20 different types of spacecraft
- * Real-time combat and docking sequences
- * Realistic trading with market fluctuations
 - * Outstanding graphics
- * Multi-screen icon based ship control systems
- * Scientifically based realistic game structure

£17.95 each

Prices all inclusive (Overseas orders add £1.00)
 Send Cheques/POs to:
GEM ELECTRONICS, 17, Tandragee Road,
Portadown, CRAIGAVON, UK. BT62 3BQ.



INTRODUCING ARM ASSEMBLER (8)

by Lee Calcraft

A roundup of ARM Instructions.

As this series draws to a close, it is an appropriate time to take stock of the areas which we have covered. To assist in this, we show the complete ARM assembler instruction set in table 1. The instructions are broken down into six groups, and you will see that some of those within the first two groups have not been treated in any detail. I hope to remedy this in some measure this month.

NEGATED OR REVERSED INSTRUCTIONS

The ARM instruction set contains four rather odd-looking instructions:

RSC	Reverse Subtract with Carry
RSB	Reverse Subtract without Carry
MVN	Move NOT
CMN	Compare Negative

These are largely provided to offset both the inflexibility in the order of expressing instruction operands, and the very limited range of immediate operands. For example, shift and rotate operations can only be applied to the rightmost operand in any instruction. Thus if you wanted to subtract a shifted register from another, you could not do it with either SUB or SBC. But RSB or RSC can be used. Both perform a subtraction with swapped operands. In other words, the two following instructions have exactly the same effect:

```
SUB R0,R1,R2
RSB R0,R2,R1
```

The first allows us to shift the contents of R2 before the subtraction (e.g. SUB R0,R1,R2,LSL #4), while the second, the contents of R1 (e.g. RSB R0,R2,R1,LSL #4).

These instructions are also useful when dealing with immediate operands (i.e. constants), because immediate values may again only be used as the rightmost operand of an instruction. You cannot therefore have an instruction of the form:

```
SUB R0,#255,R1
```

To subtract R1 from the constant 255, and place the result in R0, we must again use a reverse subtraction:

```
RSB R0,R1,#255
```

The Compare Negative instruction (CMN) works slightly differently from RSB in that it simply negates the second operand before comparing the two operands. The instruction:

```
CMN R0,R1
```

performs the equivalent of the illegal

```
CMP R0,-R1
```

The real use of this instruction is in comparing an operand with small negative numbers. This is worthwhile because of the limited range of immediate constants permitted by the ARM. You cannot for example express the number -1 as an immediate operand, and the following

Arithmetical and Logical Instructions

ADC	Add with Carry
ADD	Add without Carry
SBC	Subtract with Carry
SUB	Subtract without Carry
RSC	Reverse Subtract with Carry
RSB	Reverse Subtract without Carry

AND	Bitwise AND
BIC	Bitwise AND NOT
ORR	Bitwise OR
EOR	Bitwise EOR

MOV	Move
MVN	Move NOT

Comparison Instructions

CMP	Compare
CMN	Compare Negative
TEQ	Test Equal
TST	Test

Multiply Instructions

MUL	Multiply
MLA	Multiply and Accumulate

Branch Instructions

B	Branch
BL	Branch with Link

Register Load and Save

LDR	Load Register
STR	Store Register
LDM	Load Multiple Registers
STM	Store Multiple Registers

Software Interrupt

SWI	Perform Software Interrupt
-----	----------------------------

Table 1. The ARM Instruction Set



instruction to compare the contents of R0 with -1 would be rejected by the assembler:

```
CMPEQ R0, #-1
```

But by replacing it with:

```
CMN R0, #1
```

the instruction will assemble faultlessly.

The last of the four "odd" instructions, MVN, differs from the other three in that rather than changing the sign of one or more operands, it performs a logical NOT. Thus the following two instructions should have same effect:

```
MOV R0, #NOT &FF
```

```
MVN R0, &FF
```

though only the second would be accepted by the assembler, because NOT &FF is &FFFFFF00, and this number is not acceptable as an immediate operand. MVN can also be used to invert all the bits in a register. The following will invert the contents of R0:

```
MVN R0, R0
```

We can also use MVN to place the value -1 into R0:

```
MVN R0, #0
```

This works because inverting all the bits of the value 0 gives &FFFFFFF, and this is the two's complement representation of -1. More generally, to achieve the equivalent of:

```
MOV R0, #-n
```

you can use:

```
MVN R0, #n-1
```

FLAG AND BIT OPERATIONS

Flags can be either on or off, and may thus be efficiently represented by a single bit from a 32 bit register. Any register may thus be used to hold up to 32 independent flags, or you may choose to use a register to hold a combination of variables and flags. For example, you could hold two single-byte variables and sixteen flags all in a single register. The reason for cramming so many variables or flags into a single register is that there are only 15 registers to play with on the ARM, and although you can stack the contents of registers at any point, this always carries a time overhead.

If you are going to hold a number of flags in a single register, you will need a quick way to access each without disturbing the others. This is easily achieved using instructions from the

following group:

AND	CMPEQ
BIC	CMN
ORR	TEQ
EOR	TST

We will begin with a simple example. To set a particular bit (or flag) in R1, use:

```
ORR R1, R1, R2
```

where R2 holds a mask for the bit required. For example, if R2 holds the value 4, then bit 2 of R1 will be set as a result of the instruction, with all other bits remaining unchanged. This ORR instruction can in fact be used to simultaneously set as many bits as required, simply by placing the correct mask into R2. Thus if R2 contained the value 5, then bits 0 and 2 would be set, and so on.

To clear a particular bit, you can use the BIC instruction, again with the mask held in the rightmost operand. Thus if R3 holds the value 24:

```
BIC R1, R2, R3
```

will take the contents of R2, clear bits 3 and 4, and place the result in R1. The mask could equally well be expressed as an immediate operand:

```
BIC R1, R2, #24
```

But again the restriction on immediate operands applies, and if the mask which you require does not meet the stringent criteria applied to immediates, you will either have to use BIC repeatedly with different masks, or resort to using a mask held in a register. For example, suppose you needed to set bits 8-11 and bit 31 of register R0. The required mask is &80000F00, and this value does not fall within the subset of those acceptable as immediate constants. The solution is either to use a pair of BIC instructions with acceptable immediate operands:

```
BIC R0, R0, #&80000000
```

```
BIC R0, R0, #&F00
```

or to load the value into another register using LDR, before performing the operation. Thus:

```
LDR R4, data
```

```
ORR R0, R0, R4
```

```
:
```

```
.data
```

```
EQUED &80000F00
```



However, providing that rotation by an even number of bits would move all the data into byte zero of the word, you can specify the mask as immediate data.

When using flags, it is very convenient to give them meaningful names. This is easy to do in ARM assembler. If you have a flag at bit 5 of your flag register, say, which reflects the state of the printer, then declare a variable called *prnteron* outside the assembler listing, and give it the value 2^5 (i.e. 32). Suppose also that register 10 is being used for flags. Use the following two lines before your code is assembled:

```
flag=10
prnteron=2^5
```

Now to set the printer flag, you can use:

```
ORR flag,flag,#prnteron
```

This is far more explicit than the cryptic:

```
ORR R10,R10,#32
```

TESTING FLAGS

So far we have looked at how to set and unset specific bits in a register without affecting the remaining bits. The other operation frequently required when using flags is to test one or more bits to see whether they are set or not. The instruction TST has been provided for the purpose. It performs the equivalent of an ANDS operation on the two operands, setting the Z and N flags on the result, which is itself discarded.

To test whether our *prnteron* flag is set, we could use:

```
TST flag,#prnteron
BEQ noton
```

The function of TST is to AND the mask supplied in the second operand with the flag register in the first operand. The result of this is that the zero flag is set if the flag (or flags) is zero. But if one or more of the specified flags is set then the Z flag will be unset. Because the result of the TST operation is expressed in the state of the Z and N flags only (with C reflecting the result of the optional shift or rotate), you must take care if you are testing a group of flags. For example, if you use:

```
TST flag,#prnteron+mouse
```

to test the state of the *prnteron* and the *mouse*

flag, the Z flag will be unset if either (or both) of the two flags is set. The simplest way to obtain unambiguous information about a set of flags is to use TST once for each flag. But depending upon the particular condition which is being tested, it may be possible to take short-cuts to minimise the length of code or the execution time.

Finally, for completeness, I should mention TEQ. This instruction is similar to TST except that it uses Exclusive OR on its two operands rather than AND. Generally speaking it is only used when it is desirable to perform a comparison without disturbing the state of the carry flag. But there is a special case of both TEQ and TST when used with a P suffix when R15 is one of the operands. TEQP and TSTP can be used to set the ARM's status register without affecting the contents of the program counter. In this way the programmer can change processor modes from user mode to IRQ, Fast IRQ or Supervisor. But this is beyond the scope of the present series of articles.

Restrictions on Immediate Operands

Although both group one and group two instructions can take immediate values as their rightmost operand, only 12 bits of the 32 bit instruction word are allocated to specifying the value of such a constant. Acorn has determined that the constant should be made up of two parts - an 8-bit number (i.e. 0-255), and a 4-bit position component (i.e. 0-15). The latter determines by how many pairs of bits the number is to be rotated to the right. This means that it is possible to have constants over a much wider range than would be possible if all 12 bits had been allocated in the normal way.

Fortunately for the programmer, it is not necessary to specify constants split into their two components, the assembler does this automatically. And if the supplied constant is not able to be split in this way, the assembler will issue a "Bad immediate constant" error message.

Next month we will complete the series with a look at the Arc's assembler.



Here is our menu to whet your appetite.

ART NOUVEAU			MENU		
DISC	SHAPES	BRUSH	GOODIES	EXTRAS	TEXT
Load Save Use new disc Change Drive Initialise Delete file	Filled in In outline Single line Rubber line Radii line Rectangle Square Parallelogram Triangle Ellipse Circle Polygon Arc Sector Segment User	Use brush Pickup Choose Transparency Size Flip Special effect	Choose colour Choose pattern Flood fill Drawing action Pencil Spray brush Pixel editor Pattern editor Grid Window	Mouse speed Printer dump Colour change Colour merge Move screen Lock mouse Colour cycle Scratch screen	Print text Font editor

Only £42.50 inc. VAT and p+p

The above menu is served with;

- * 256 colours.
- * Facility to load MODE 15 or MODE 12 screens from other art packages.
- * Facility to load and save standard or compressed screens.
- * User defined irregular shapes.
- * Facility to Shear, Band, Wave and Squiggle the brush.
- * Spray with points, circles, squares or brushes.
- * User defined spray area.
- * A screen grid
- * Colour selection from palette or from screen.
- * Facility to change mouse working area.
- * Colour merging.
- * Screen scrolling.
- * Facility to load and save brushes, patterns and colour cycles.
- * 8 x 32 colour cycles.
- * Facility to define transparent colours when overlaying.
- * Colours which can be set to AND, OR, EOR, INVERT.
- * A pixel editor with magnification of up to 16 times.
- * A pattern editor with up to 32 different patterns per file.

plus many more, making a total of over 100 different selections available!

Art Nouveau

THE new ART PACKAGE for the Archimedes by Barry Christie.

Are you ready to order now?



Order now on our 24-HOUR ACCESSCREDIT CARD HOTLINE on (0698) 733775 or send a cheque for £42.50 made payable to 'COMPUTER ASSISTED LEARNING LTD', at

COMPUTER ASSISTED LEARNING LTD. (Dept RU),
Strathclyde Business Centre,
Princess Road,
New Stevenston ML1 4JB



MATRIX-3

SCHOLAR OR SCIENTIST FINANCIER OR PHYSICIST
MATRIX-3 ADDS A NEW DIMENSION TO PROBLEM SOLVING
THE 3D SPREADSHEET FOR THE ARCHIMEDES RANGE (1Meg min)

Features include:

75 Functions 30 Commands
Cell programs
Evaluate/Set cell
Optimised calculation
NEW - CSV files supported
Allows graphics via PRESENTER
(Available from Linguinity)



19 Panton Street, Cambridge
CB2 1HL Telephone 0223 66553

£95.00 + VAT

To order: Send cheque for
£109.25 payable to
Cambridge Microsystems
Official orders accepted
Site licences available
Phone or write for brochure

ARCHIMEDES SOFTWARE

- Disc 1 EMACS Multiple buffer/document UNIX super editor with integral programming language, programs, tutorial, and manual
- Disc 2 Micro Spell 43,000 word spell checker. Ideal for EMACS. New memory resident module with continuous spell check.
- Disc 3 Fortune Cookie. Database of over 7000 amusing quotations.
- Disc 4 XLISP Object oriented version of LISP with C source code.
- Disc 5 C Toolkit. 20 programs including grep, awk, sed, ed, make, tall, cross ref., pretty print, file compare, sort, join, split, etc.
- Disc 6 Kermit comms/file transfer program. Better than Acam Kermit.
- Disc 7 Chess. A good chess game running under the Wimp environment. Lots of features; save & load games, edit board, computer play.
- Disc 8 Cross Stor. Wimp based crossword puzzle solver with massive dictionary. Makes solving and compiling puzzles easy.
- Disc 9 File Tools. arc, compress, uuencode, atab, strings, cut, paste, fgrep.
- Disc 10 More C Tools. yacc & lex (with src. & examples), diff, ctags etc.

Each disc is £5.99 inc. Buy four claim one free!

**Available from David Pilling,
P.O. Box 22, Thornton Cleveleys, Blackpool. FY5 1LR.**

Reviewed by Mark Sealey

When I looked at EMR's earlier music package, Soundsynth (RISC User Volume 1 Issue 7), I explained that the product would form part of a larger and very ambitious suite of programs for the Archimedes, called Arpeggio. True to their word, EMR has now released a sequencing package, Studio 24 Plus, which will allow composition of tunes and rhythms as opposed to the generation of waveforms performed by Soundsynth.

The criteria used for evaluation in this review are ease and flexibility of use on the one hand and features and facilities on the other.

MIDI

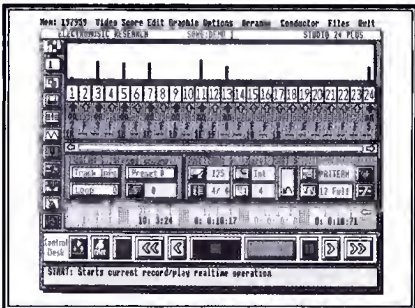
It goes almost without saying, that Studio 24 Plus relies entirely on the MIDI standards, which EMR's Mike Beecher has done so much to establish, particularly where Acorn machines are concerned. Indeed, Studio 24 Plus will not produce any sounds without a MIDI hardware interface fitted. It also requires at least a 1 Mbyte machine.

MIDI (Musical Instrument Digital Interface) originated in an attempt to allow instruments making use of computers to work together. The highly successful MIDITRACK series of components which EMR produced for the earlier BBC models shows just what can be achieved. Now the same integrated scheme is taking shape for the much more versatile and powerful RISC machines. And a good thing too! This far, EMR looks to have cornered the market, so their software had better be good!

STUDIO 24 PLUS

What is being reviewed here is officially termed a MIDI Recording Sequencer. This implies, at its simplest, that you can compose and control a sequence of notes or chords, edit them, and then hear them using any of the waveforms supplied by Acorn (Stringlib, Percussion etc.), or (better) those created using Soundsynth either directly or from the 35 examples contained on the *Creations Sound Disc* (£19.95 from EMR).

If that were all Studio 24 Plus could do, there wouldn't be much to recommend it. In fact it stretches sound production on a personal computer to a degree of sophistication which is almost unprecedented. As such, Studio 24 Plus deserves to sell and sell.



USING THE SOFTWARE

On booting the protected disc, you have the opportunity to change the default allocation of memory for the WFS (Waveform Filing System - roughly equivalent to older BBC Basic's ENVELOPE command - see Soundsynth review) before a digitised screen image of a keyboard and score follows. This gives way to the main control screen of the program. The first impression is of an overcrowded and cluttered cockpit, with dials and arrows everywhere.

This is quickly dispelled. In the first place, many of the icons are already familiar and conventional ones: cassette recorder PLAY, RECORD and FAST FORWARD etc. Secondly, the other icons are well-designed, and their meanings are largely self-evident. The tuning-fork sets the exact internal Archimedes pitch, for example.

The package's designer has chosen deliberately not to go for a system which sends you down multiple menu trees to obtain all the effects incorporated. Everything has to be

STUDIO 24 PLUS

visible at the same time. This is a brave (and risky) decision. In practice, though, it works. A few minutes will actually accustom the user to moving around the options. The manual is just clear enough to be useful (though dense in its present form), but Artisan-style help text in a two line window at the foot of the screen doesn't go far enough.

As can be seen from the illustration, the top line is a menu bar, controlling the program's main functions. Some of these options open non-WIMP boxes which are clear and easy to use even if lacking a little aesthetically by the standards of some current Archimedes software.

On the left is a vertical row of icons which controls music-making. The centre of the screen both reflects what is happening and controls tracks and patterns etc., as well as displaying the music itself at the top in one of two ways.

The use of the three mouse buttons is consistent and easy to learn, as are the techniques for locating any position in any track; this could have been very confusing with so much going on.

Once the overall structure of the package becomes clear (and the manual again will come to your rescue if necessary), composing and editing music quickly becomes a delight. It can be done using a MIDI instrument, which is really the only sensible approach in the long run, or the computer's own keyboard. The results are best heard by connect MIDI output to a hi-fi stereo system. It needs to be said that, when all this is working, the results are very pleasing, as anyone who has experienced one of Mike Beecher's own demonstrations will know!

If you are still unsure about how to get started, there is a five-page tutorial-style section to take you step-by-step through recording and playing back a simple tune. Any experience you have of Acorn's Music Editor on the Archimedes Welcome Disc will only be

vaguely relevant, as Studio 24 Plus is so much more advanced and intricate in its functions.

FEATURES

In the first place, this is a MIDI package and as such makes full use of multiple channels. Studio 24 Plus version 1.0 (this one) utilises up to 64 of them, which gives you real synthesiser power. In the old days the same channel was normally used by each instrument for its data. One strength of this package is its potential for accessing over 8,000 MIDI channels (via the SMPTE podule).

Up to four IN ports can be accepted simultaneously as well. A professional version of the software is planned to expand this even further. The permutations at present, though, ought to be enough to keep anyone going. With Studio 24 Plus, MIDI IN and OUT are both controllable, and different instruments (or multi-timbral instruments) can be distributed across different tracks. As explained previously, to hear what music is being produced it is necessary to have either EMR's MIDI-4 Interface Expansion card, or Acorn's own MIDI Interface, but not the MIDI add-on to the Acorn Input-Output extension (see RISC User Volume 1 Issue 5 p.24).

As a result of all this, you could have many different instruments playing simultaneously at various locations in the stereo spectrum on many tracks. Completed compositions or work in progress can be saved to and loaded from disc of course. This is achieved by replacing the system disc (not normally needed after booting) with one, or several, data discs to store music. This all works smoothly. One of the indicators on the menu bar is of free memory in the Archimedes; some idea of eventual disc space to be occupied by each composition would have been useful too.

The following details may be of somewhat specialist interest, and concern the variations in sound which Studio 24 Plus permits. Overall, the possibilities here are considerable, well-implemented and likely to meet the requirements of most users.

TRACKS AND PATTERNS

To the newcomer, MIDI data seems to be a complicated hierarchy of component items. Perhaps it is useful to draw an analogy with text in a word processor, where ASCII characters, words, lines, blocks and whole files can all be addressed. So Studio 24 Plus allows full manipulation of MIDI *events*, *tracks*, *patterns* and *songs*.

These are explained reasonably clearly in the manual, although some examples would have been helpful. Studio 24 Plus does deal in *events* (a single note consists of a *note on* and *note off* event together with a representation of its channel and pitch etc.). A sequence of these makes up a *track*; each track is both polyphonic and at the same time discrete in that it can be treated separately from other tracks. Don't think of tape-tracks, though. A *pattern* is probably best thought of as a packet of several tracks whilst a *song* is the complete collection of sequenced patterns regardless of their order of composition.

A good sequencer needs to make manipulation of these elements, as well as their relationship to one another, crystal clear. The difference between software that does and does not can be enormous, but Studio 24 Plus sweetens, rather than obscures, the process.

Facilities particularly worthy of mention are an easy-to-use *Editor* to monitor, change, copy, cut and paste sections of any data; control of tempo achieved in a variety of ways (time signatures, metronomes etc.), and a *Conductor* feature, which is a special, invisible track, to cue and synchronise tempo. Sequences can be transposed, and a transpose-offset feature allows composition at one pitch to sound at another - all to make things easy for the user.

EMR has used Studio 24 Plus with primary age children, and it is not hard to see how well this could work. Suffice it to say that for a hundred pounds, you have the heart of a very sophisticated and yet easy to use piece of software.

EXPANSION

Studio 24 Plus may well, eventually, be used as part of an even larger system, for instance, to control stage lighting or, with EMR's forthcoming Scorewriter, to print musical scores on a laser printer. Their MusicScan will also read sheet music for feeding into the system. If used with a sound sampler (such as Armadillo's, see RISC User Volume 1 Issue 5, and later, stereo versions), or in sync with a visual image digitiser, professional quality work is well within the grasp of anyone who has the imagination.

CONCLUSIONS

This product has much in it, almost too much, maybe (some fifty features are listed in the specification). At times, a complete beginner may feel daunted by this. But because there is instant audio feedback, learning is easy. Because so much can be achieved early in the 'learning curve', you are encouraged to go on and see where the refinements fit in. What is more, potential sources of confusion, such as the manipulation of multiple tracks and allocation of multiple MIDI channels, are avoided more by good overall structural design of the software and the screen than by the manual, which is not over generous with its examples, pace or definitions.

As with text processing, sizeable amounts of data have to be scrolled to be viewed and edited. Unlike text, the nature of MIDI information means that this happens in more than one dimension, as if coping with simultaneous translations and etymology of words at the same time as conventional word processing. The way this works without ever losing even a tired reviewer in a welter of notes is to the credit of this lively software house. Thoroughly recommended.

**Product
Supplier**

Studio 24 Plus
EMR (Electro Music Research)
14 Mount Close,
Wickford, Essex SS11 8HG.

Price

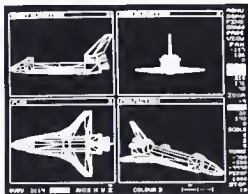
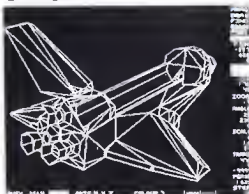
Tel. (0702) 335747
£99.00 inc. VAT.

RU

SILICON VISION

SOFTWARE FOR THE ARCHIMEDES & BBC

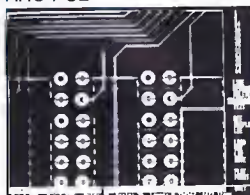
SolidCAD



The ultimate 3D graphics system for the Archimedes. Architectural design, Interior design, Engineering and Freeform CAD. Allows drawing in plan, front & side elevations and also directly in perspective. Includes powerful zoom & pan options for precision draughting and surface shading. Creating Solids about objects. Also includes Sweep, Extrude & Macro commands for designing very complex objects easily. Designs created with SolidCAD are compatible with the Realtime Graphics Language for high speed flicker free animation. The Archimedes version also performs smooth shading for realism. Both ADAs users can upgrade to the Realtime Solids Modeller (Arc) for £40.00

£49.95 (ARC & BBC) **New**

ARC-PCB



The ultimate PCB design system developed specifically for the Archimedes with a specification that cannot be matched. Includes Automatic routing. Rats nesting, 8 layers, Surface mount capability 0.001 resolution 32 x 32 maximum board size, On-line Help, Fast Zoom Pan Redraw, Text & Silkscreen facility. Variable Line Pad Text Grid sizes. Part Libraries, Block Move Copy Rotate Mirror Erase options, and up to 300 000 components. For hardcopy the system supports a large number of plotters and ordinary Epson compatible printers at very high resolutions (1920 x 1024) of 240 dots/inch for near laser quality output. An entry level ARC PCB system (without auto routing) is available for £99.95 (Auto routing upgrade £100). Enquire about our turnkey PCB design systems complete with Colour Archimedes and/or plotters with prices from £1220 to £3809 (inc. VAT)

£195.00 (ARC) **New**

REALTIME SOLIDS MODELLER

The Realtime Solids Modeller includes both the sophisticated design environment of SolidCAD and the high speed animation capability of the Realtime Graphics Language (RGL) module developed in pure ARM Risc code for supercharged performance. The package is ideal for architectural design, Interior design, Engineering design & teaching COT. The RGL module can be used to create standalone flicker free animation of designs from your own models. Smooth shading is also performed for realistic images. Through our in-house Realtime CAD Design and High speed techniques, no other package can rival the design speed & animation speed of the Realtime Solids Modeller

£89.95 (ARC) **New**

REALTIME GRAPHICS LANGUAGE

The Realtime Graphics Language provides a complete 3D Solids Wireframe animation system with a set of commands and 3D Editors for designing objects. It is to animate wireframe models at 3000 pixels/sec line generator for fast 3D drawing. Includes a 3D Print Perspective and Turbographics. Also compatible with the Archimedes and BBC

£49.95 (BBC)

SUPER-DUMP

Super-Dump is the only software which takes advantage of the highest resolution capability of the Archimedes to provide 1920 x 124 resolution images can also be previewed before printing. Fully compatible with SolidCAD. Realtime for the Archimedes. Gate Array design system & 3D CAD Animation system. Viewport 2400 x 1200 pixels. All packages can be made compatible with Super-Dump. An example program is included. An example program is included. The

£15.95 (BBC), £24.95 (ARC) **New**

Presentation Manager

Presentation Manager is a software package for the Archimedes. It is designed to create presentations. Also for the BBC. It is designed to create presentations. Also for the BBC. It is designed to create presentations. Also for the BBC.

£34.95 (BBC), £49.95 (ARC) **New**

SILICON VISION LTD, SIGNAL HOUSE, LYON ROAD, HARROW MIDDLESEX HA1 2AG. TEL: 01-422 2274 or 01-861 2173

FAX: 01-427 5169. TELEX: 918266 SIGNAL G.

(Access/Mastercard/Eurocard accepted)

All prices include VAT and Carriage (Overseas orders add £4).

RiscBASIC

Supercharge your Archimedes programs by compiling them automatically into pure ARM Risc with the RiscBASIC compiler. Features include Relocatable modules, Cross reference of all variables, Function procedures, Floating point arithmetic support. Stand alone code generator. Optimising compiler & Full runtime handler

£99.95 (ARC) **New**

RiscFORTH

A new 32 bit implementation of the FORTH standard designed to take advantage of the ARM architecture. Features include Multi-tasking, Optimising compiler built in ARM assembler with floating point routines, built in FORTH interpreter, Full system integration, it's support for the Archimedes & other ARM based systems. Stand alone code generator. Optimising compiler & Full runtime handler

£99.95 (ARC) **New**



Postbag

We welcome your letters for publication on our Postbag page whether they contain comments on the magazine and the Archimedes, technical queries or information for other readers.

HIGHLY VISUAL

After typing in the Swirl program from Volume 2 Issue 1 (Archimedes Visuals), and replacing lines 210 and 220 as suggested, I thought that the resulting display looked a little lumpy where the edges of the ellipses were visible. I found that by increasing the number 32 in line 210, this solved the problem. A better effect is created by changing line 210 to read:

```
210 ELLIPSE FILL 640-N%,512-N%,200,840,  
PI*N%/160
```

Timothy Candy

Many of the 'visuals' which we publish are ideally suited for experimenting with to produce a variety of new effects, and we are always happy to see the results of members' efforts. Thanks to Mr.Candy for the above suggestions.

THANKS FOR THE PROGRAM

Congratulations to David Spencer for his fine effort with his program "Toolbox". We could do with more long programs like this.

R.Parsons

CLARIFYING ARM ASSEMBLER

I have been following Dr. Calcraft's series *Introducing ARM Assembler* with great interest, but have two points of confusion. I wonder if you could clarify these for me please.

In the first two articles there is a line of code in the example program:

```
210 EQU00:EQU00:EQU00
```

Quite simply what does this do? I also gather that P% and Q% contain particular values after the code has been assembled. could you tell me what these are?

M.Tiltmas

The EQU0 instruction is one of a class of instructions known as pseudo-ops. Rather than assembling to an ARM machine code instruction, pseudo-ops tell the assembler to do something. In the case of EQU0, the

*following value, zero here, is stored in memory as four bytes (one word). This allows numeric values to be included in the middle of assembler programs. EQU0, and the other pseudo-ops available (such as OPT and EQU5), will be covered in the final part of *Introducing ARM Assembler* next month.*

*The values contained in P% and Q% control where the assembled machine code will be stored, and where it will be run from. Normally, only P% is used. This is set to the address at which to store the machine code before assembly starts, and is incremented automatically as assembly proceeds. When assembly is complete, P% will point to the next memory address after the machine code. This is useful when working out the end address of an assembled program, for example when saving it. By using Q%, the machine code can be stored in memory at a different place to that from which it is designed to run. This topic will also be covered when the final part of *Introducing ARM Assembler* looks at Basic's built-in assembler.*

WHITHER RISC USER?

I enjoy your magazine but find some parts very technical. I (and my children) would very much welcome some fairly simple articles for people whose first home computer this is - though I have used IBMs or compatibles at work. We are keener to use the machine than to do very clever things to files.

Barbara Wilson

As a result of initial resesarch we deliberately included a high level of technical content in RISC User. It is likely that many of the first purchasers of the Archimedes were enthusiastic Beeb or Master owners who wanted to upgrade to the faster and more technically advanced Archimedes. Now we are receiving an increasing number of letters from people such as Mrs Wilson for whom the Archimedes is their first Acorn computer (indeed, possibly their first computer purchase ever). Although we intend to continue to maintain a well researched technical content, we shall be endeavouring to provide more information for the more application oriented user, for example our coverage of star commands in this issue. If you have any comments or suggestions for the contents of future issues of RISC User then please let us know.

RU

BACKUP IN BROMLEY

AUTHORISED ACORN DEALERS



We offer full technical backup, service and advice as well as a comprehensive range of software and hardware for your Archimedes. If you have a problem - we can help you.

Call in for a software or hardware demonstration,
or phone for prices and availability

Data Store

6 Chatterton Road, Bromley, Kent
Telephone : 01-460 8991 (closed Wednesdays)

ARCHEFFECT

FX

VISUAL EFFECTS
MODULE

ARCHEFFECT is an easy to use image manipulation package. By the use of simple 'Commands' you will be able to quickly produce spectacular visual effects, similar to those seen on TV. It also allows for the easy use of fonts.

The module has been specifically designed for use in the high resolution graphics modes to take full advantage of the potential of the Archimedes.

In addition to the many image manipulation commands the module provides an alternative method of using the powerful font manager provided on the Archimedes, bypassing the complex 'SYS' commands previously necessary.

ARCHEFFECT - £24.95 inc VAT and p&p

Please send me copies of Archeffect at £24.95 each.

I enclose a cheque or postal order for the sum of £.....

Please make cheques or postal orders payable to FX.
FX, 207 South Avenue, Southend-on-Sea, Essex SS2 4HT.
(For further details send SAE)

Another collection of useful information rounded up by David Spencer.

COMPATIBLE CALLS

Basic V checks the addresses given as arguments to CALL and USR, and if they constitute a valid Beeb entry point in the range &FFCE to &FFF7, Basic will emulate the appropriate Beeb call. This means that rather than calling that address in memory, the appropriate SWI is executed. For example:

```
A%=65
```

```
CALL &FFEE
```

will print the letter 'A' on the screen in the same way as:

```
SYS "OS_WriteC",65
```

This attempt at compatibility can cause problems if you try to call a genuine routine in this area of memory. However, the only Beeb entry points that clash with valid word-aligned ARM addresses are &FFD4, &FFE0, and &FFF4. These can either be avoided, or you can give a dummy parameter to the CALL statement, for example:

```
CALL &FFF4,A
```

because if any parameters are present the Beeb emulation is bypassed.

FUNCTION KEYS

In addition to the twelve red keys on the Archimedes, the keys marked 'Print' and 'Insert' are also treated as function keys. The 'Insert' key is key number 13, and the 'Print' key is key number 0. This means that old Beeb software that used function key f0 can still be used on the Archimedes, because 'Print' will behave in the same way as f0.

QUICK COLOUR SCALES

The anti-aliased font handling provided by the Archimedes offers a very quick way of setting up a gradation of colour between two RGB values. There are a couple of limitations though. Firstly, the number of colours to use must be either 2, 4, 8 or 16, and secondly, it will not work in 256 colour modes. The command to use is:

```
VDU 23,25,n|
```

```
VDU 23,25,s+&80,s+1,r1,g1,b1,r2,g2,b2
```

The value of *n* determines the number of colours, and this is 1 for two colours, 2 for four

colours, 3 for eight colours, or 4 for sixteen colours. The scale starts at colour number *s*, and fades from the colour given by *r1,g1,b1* to that of *r2,g2,b2*. For example:

```
VDU 23,25,3:23,25,&88,9,255,0,0,0,0,255
```

in mode 12 would change colours 8 to 15 to a fade between red and blue.

SPRITE COLOURS

In Arthur 1.20 the call to SWI "OS_SpriteOp" to read the colour of a pixel within a sprite doesn't return the correct value in 256 colour modes. The tint number for the pixel is however returned correctly - for what it's worth. It is because of this bug that the Welcome Sprite Editor (SEdit) does not display colours correctly in 256 colour modes.

VERTICAL BAR IN VDU

One very useful feature of the VDU statement in Basic V is the ability to terminate it with a vertical bar character (|). This causes nine zeros to be sent to the VDU driver. As no VDU statement requires more than nine parameters, a '|' will always terminate the current command. For example:

```
VDU 23,17,5|
```

swaps the text foreground and background colours. If less than nine zeros are needed to finish the current command, then the extra ones are treated as NUL characters and ignored.

The vertical bar doesn't have to be at the end of the VDU command. For instance:

```
VDU 23,17,5|ASC"A",23,17,5|
```

will invert the colours, print a letter 'A', and then revert the colours to normal. The '|' is used here as a separator just as ',' or ';' would be.

A TINT OF TROUBLE

The TINT keyword in Basic can be used in three different ways. Firstly, TINT can be appended to COLOUR or GCOL to set the tint at the same time as the colour. Secondly, TINT can be used as a function in the form:

```
<variable>=TINT(X,Y)
```

in which case the tint of the pixel at co-ordinates (X,Y) is returned. The third use, and

HINTS & TIPS

the one that many people are unaware of, takes the form:

TINT type,tint
which sets the tint of the current background or foreground, graphics or text colour, depending on the value of **type**:

type=0 Text foreground tint
type=1 Text background tint
type=2 Graphics foreground tint
type=3 Graphics background tint

The value of **tint** is 0, 64, 128 or 192 as normal.

VOICING AN OPINION

The *Archimedes User Guide* fails to mention the Basic statement **VOICE**, which is used to define the voice to be used for a particular sound channel. The syntax is:

VOICE <channel>,<voice>

where **channel** is a number from 1 to 8, and **voice** is a string containing the name of the voice. For example:

VOICE 1, "Percussion-Soft"

HINTS & TIPS

The voice name must be exactly as it is listed by ***VOICES**.

PC DELETE

When using wildcards in the **DEL** command on the PC Emulator, for example:

DEL my?r*.wk?

it is very easy to delete more than you bargained for. A clever way of checking that you are actually deleting the correct files is to first of all type:

DIR my?r*.wk?

which lists all the files that match the wildcard name (in this case 'my?r*.wk?'). You can check that the list is what you had expected, and if it is, type **DEL** and press function key F3. This recalls the rest of the previous command line, which in this case is the filename. Pressing Return will now erase all the files that were listed. Thanks to Mr. C Marlow for this tip.

RU

Technical Assistant

BEEBUG needs a Technical Assistant to work in our magazine department to provide technical support for both BEEBUG and RISC User.

The successful candidate should have a good knowledge of programming in Basic, and preferably an understanding in an Assembler. Experience of using these languages on a BBC micro is highly desirable. The work will involve dealing with technical queries relating to our magazines, evaluating contributions, writing, editing and testing programs for use internally or for publication, and generally providing technical expertise and support.

If you are interested in working for a friendly but go-ahead organisation in an interesting and challenging job, then we would like to hear from you as soon as possible.

To apply, write to The Personnel Manager at the address below, giving full details of yourself, your education and work experience (if any), and convince us you are the right person for the job.

BEEBUG Ltd, Dolphin Place, Holywell Hill, St Albans, Herts AL1 1EX.

OVID TOOLKIT MODULE

provides the following facilities:-

*MEDIT	memory editor
*DEDIT	disc editor 800 & 640K
*DIS	disassembler
*DSIZE	shows size of disc
*SHIFT	shifts memory
*FILLB	fills memory with a byte
*FILLW	fills memory with a word
*COMP	compares two memory areas
*MFINDSCS	finds a (wild carded) string in memory (case sensitive)
*MFINDSCI	finds a (wild carded) string in memory (case insensitive)
*MFINDW	finds a (wild carded) word in memory
*DFINDSCS	finds a (wild carded) string on disc (case sensitive)
*DFINDSCI	finds a (wild carded) string on disc (case insensitive)
*DFINDW	finds a (wild carded) word on disc

Send a cheque/PO for £15 to:-

P.A.J. Taylor, 23 Limes Ave, Afreton, Derbys DE5 7DW.

RISC User Magazine Disc

December 1988

ARCHIMEDES VISUALS. Two more graphical programs. The first allows the easy choice of colours in 256 colour mode by using a three-dimensional grid, while the second illustrates the use of masks for sprite plotting.

MOUSE MENU

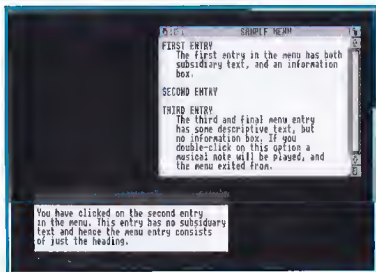
A WIMP based universal menu system. As an illustration of its power a complete menu to run any program from this disc is given.

MOVIE MAKER

A program to design and animate a sequence of frames. A sample cartoon is included to experiment with.

RISC USER TOOLBOX

The Toolbox is extended to allow the printing of memory dumps and disassembled listings. Also included is a file comparison utility and an intelligent ADFS disc compacter.



* BONUS ITEMS *



MORIC

When you get bored at Christmas play this highly addictive game.

STUDIO 24 SCREENSHOT

A screen shot from EMR's *Studio 24* system reviewed in this issue.

RISC OS SCREENSHOT

A sample of the RISC OS desktop to show the much improved appearance.

NEW RISC USER DISC MENU

A version of last month's coloured disc menu that will work on unnammed discs.

ARCSCAN DATA

Bibliographies for this issue of RISC User and the latest BEEBUG (Vol.7 No.6) for use with Arcscan.

RISC User magazine discs are available to order, or by subscription. Full details of prices etc. are given on the back cover of each issue of RISC User.



RISC User BINDERS

£3.90 + £1 p&p

BEEBUG Ltd
Dolpin Place, Holywell Hill, St. Albans, Herts
AL1 1EX
Tel. (0727) 40303

RISC USER magazine

MEMBERSHIP

RISC User is available only on subscription at the rates shown below. Full subscribers to RISC User may also take out a reduced rate subscription to BEEBUG (the magazine for the BBC micro and Master series).

All subscriptions, including overseas, should be in pounds sterling. We will also accept payment by Connect, Access and Visa, and official UK orders are welcome.

RISC USER SUBSCRIPTION RATES

£14.50	1 year (10 issues) UK, BFPO, Ch.I
£20.00	Rest of Europe & Eire
£25.00	Middle East
£27.00	Americas & Africo
£29.00	Elsewhere

RISC USER & BEEBUG

£23.00
£33.00
£40.00
£44.00
£48.00

BACK ISSUES

We intend to maintain stocks of back issues. New subscribers can therefore obtain earlier copies to provide a complete set from Vol.1 Issue 1. Back issues cost £1.20 each. You should also include postage as shown:

Destination

First Issue
Each subsequent Issue

UK, BFPO, Ch.Is

60p
30p

Europe plus Eire

£1
50p

Elsewhere

£2
£1

MAGAZINE DISC

The programs from each issue of RISC User are available on a monthly 3.5" disc. This will be available to order, or you may take out a subscription to ensure that the disc arrives at the same time as the magazine. The first issue (with six programs and animated graphics demo) is at the special low price of £3.75. The disc for each issue contains all the programs from the magazine, together with a number of additional items by way of demonstration, all at the standard rate of £4.75.

MAGAZINE DISC PRICES

	UK	Overseas
Single issue discs	£ 4.75	£ 4.75
Six months subscription	£25.50	£30.00
Twelve months subscription	£50.00	£56.00

Disc subscriptions include postage, but you should add 60p per disc for individual orders (30p for additional discs on the same order).

All orders, subscriptions and other correspondence should be addressed to:

RISC User, Dolphin Place, Holywell Hill, St Albans, Herts AL1 1EX.

Telephone: St Albans (0727) 40303

(24hrs answerphone service for payment by Connect, Access or Visa card)